Package ‘refineR’

May 9, 2026

Type Package

Version 2.0.0

Date 2025-11-05

Encoding UTF-8

Title Reference Interval Estimation using Real-World Data

Author Matthias Beck [aut, cre] (ORCID: <`https://orcid.org/0009-0008-7678-4581`>), Tatjana Ammer [aut] (ORCID: <`https://orcid.org/0000-0003-2809-0266`>), Christopher M Rank [aut], Andre Schuetzenmeister [aut] (ORCID: <`https://orcid.org/0000-0002-2964-5502`>)

Maintainer Matthias Beck <`matthias.beck.mb1@roche.com`>

Depends R (>= 3.2.0)

Imports stats, ash, future, future.apply, parallel, graphics, grDevices

Suggests knitr, rmarkdown

Description Indirect method for the estimation of reference intervals (RIs) using Real-World Data ('RWD') and methods for comparing and verifying RIs. Estimates RIs by applying advanced statistical methods to routine diagnostic test measurements, which include both pathological and non-pathological samples, to model the distribution of non-pathological samples. This distribution is then used to derive reference intervals and support RI verification, i.e., deciding if a specific RI is suitable for the local population. The package also provides functions for printing and plotting algorithm results. See ?refineR for a detailed description of features. Version 1.0 of the algorithm is described in 'Ammer et al. (2021)' <doi:10.1038/s41598-021-95301-2>. Additional guidance is in 'Ammer et al. (2023)' <doi:10.1093/jalm/jfac101>. The verification method is described in 'Beck et al. (2025)' <doi:10.1515/cclm-2025-0728>.

License GPL (>= 3)

VignetteBuilder knitr, rmarkdown

NeedsCompilation no

LazyData TRUE

RoxygenNote 7.3.2

Repository CRAN

Date/Publication 2025-11-06 17:20:02 UTC

Contents

`refineR-package` *refineR: Reference Interval Estimation using Real-World Data (RWD)*

Description

This package includes the implementation of the refineR algorithm (Ammer et al., 2021) which is an indirect method for the estimation of reference intervals using Real-World Data (RWD). It takes routine measurements of diagnostic tests, containing pathological and non-pathological samples as input and uses sophisticated statistical methods to derive a model describing the distribution of the non-pathological samples. This distribution can then be used to derive reference intervals. Version 2.0.0 added functions to check whether a reference interval is valid for the population from which the RWD originates. Main function of this package is `findRI` that takes an input data set and tries to find a model that best explains the non-pathological distribution. Furthermore, the package offers functions for printing `print.RWDRI` and plotting `plot.RWDRI` the results of the algorithm operating on S3-objects of class 'RWDRI'. Verification is performed with the `verifyRI` function.

Details

Package:	refineR
Type:	Package
Version:	2.0.0
Date:	2025-11-05
License:	GPL (>=3)
LazyLoad:	yes

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>, Tatjana Ammer <`tatjana.ammer@roche.com`>, Christopher M Rank <`christopher.rank@roche.com`>, Andre Schuetzenmeister <`andre.schuetzenmeister@roche.com`>

References

Ammer, T., Schuetzenmeister, A., Prokosch, H.U., Rauh, M., Rank, C.M., Zierk, J. (2021). refineR: A Novel Algorithm for Reference Interval Estimation from Real-World Data. Sci Rep 11, 16023. doi:10.1038/s41598021953012 Ammer, T., Schützenmeister, A., Rank, C. M., Doyle, K. (2023). Estimation of Reference Intervals from Routine Data Using the refineR Algorithm-A Practical Guide. (2023). J Appl Lab Med 8 (1), 84-91. doi:10.1093/jalm/jfac101 Beck, M., Dufey,

F., Ammer, T., Schützenmeister, A., Zierk, J., Rank, C.M., Rauh, M. (2025). VeRUS: verification of reference intervals based on the uncertainty of sampling. Clin Chem Lab Med. doi:10.1515/cclm20250728.

`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`>

`alignVec` *Helper function to align the content of printed tables*

Description

This function aligns vectors by padding with spaces to the right

Usage

```
alignVec(x, digits = 3)
```

Arguments

`x`	(numeric or character) vector of values to be aligned
`digits`	(integer) number of digits to be displayed for numeric values

Value

`(character) vector of aligned values`

Author(s)

Christopher Rank <`christopher.rank@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 completey transparent and 1 meaning completey 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)
```

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`>

Examples

```
## Not run:
# convert character string representing a color to RGB-code using alpha-channel of .25 (75\
     as.rgb("red", alpha = .25)

# same thing now using the RGB-code of red (alpha=1, i.e. as.rgb("red"))
     as.rgb("#FF0000FF", alpha = .25)

## End(Not run)
```

`ashDensity` *Estimate density of distribution employing the R package "ash" using R-wrapper function.*

Description

Estimate density of distribution employing the R package "ash" using R-wrapper function.

Usage

```
ashDensity(x, ab, nbin, m, kopt = c(2, 1), normToAB = FALSE)
```

Arguments

`x`	(numeric) vector of data points
`ab`	(numeric) vector of lower and higher truncation limit of density estimation
`nbin`	(integer) specifying the number of bins used for density estimation
`m`	(integer) specifying the width of the smoothing kernel(s) used for density estimation
`kopt`	(integer) vector specifying the smoothing kernel
`normToAB`	(logical) specifying if the density is normed to the interval ab or to all data points in x

Value

(list) with density estimation (x values, y values, m and ab).

Author(s)

Christopher Rank <`christopher.rank@roche.com`>, Tatjana Ammer <`tatjana.ammer@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`>

`calculateCostHist` *Calculate costs for a specific combinations of lambda, muVec and sigmaVec.*

Description

Calculate costs for a specific combinations of lambda, muVec and sigmaVec.

Usage

```
calculateCostHist(
  lambda,
  muVec,
  sigmaVec,
  HistData,
  alpha = 0.01,
  alphaMcb = 0.1,
  pNormLookup
)
```

Arguments

`lambda`	(numeric) transformation parameter for inverse Box-Cox transformation
`muVec`	(numeric) vector of mean values of non-pathological Gaussian distribution in transformed space
`sigmaVec`	(numeric) vector of sd values of non-pathological Gaussian distribution in transformed space
`HistData`	(list) with histogram data generated by function `generateHistData`
`alpha`	(numeric) specifying the confidence region used for selection of histgram bins in cost calculation
`alphaMcb`	(numeric) specifying the confidence level defining the maximal allowed counts below asymmetric confidence region
`pNormLookup`	(list) with lookup table for pnormApprox function `pnormApprox`

Value

(numeric) vector with (lambda, mu, sigma, P, cost).

Author(s)

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

`cdfTruncatedBoxCox` *Function to calculate the RI from a set of parameters*

Description

Function to calculate the RI from a set of parameters

Usage

```
cdfTruncatedBoxCox(mu, sigma, lambda, shift, RIperc)
```

Arguments

`mu`	(numeric) mean of the distribution
`sigma`	(numeric) standard deviation of the distribution
`lambda`	(numeric) Box-Cox transformation parameter
`shift`	(numeric) shift of the distribution
`RIperc`	(numeric) vector of percentiles for which the reference interval should be calculated

Value

(numeric) vector of reference interval values for the given percentiles

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>, Christopher Rank <`christopher.rank@roche.com`>, Tatjana Ammer <`tatjana.ammer@roche.com`>

References

Freeman, R. Modarres / Statistics & Probability Letters 76 (2006) P 767

`checkInvalidArgs` *Check Invalid Arguments*

Description

This function checks if the arguments in param are valid

Usage

```
checkInvalidArgs(param, validArgs, marginType, verbose)
```

Arguments

`param`	(list) providing non-default parameters for the calculation of the margins
`validArgs`	(character) vector of valid arguments for the calculation of the margins
`marginType`	(character) specifying calculation of the margins: using "VeRUS", or Equivalence Limits ("EL")
`verbose`	(logical) specifying if additional warning messages are printed

Value

warning message if invalid arguments are found

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>

`createVerificationTab` *Create Verification Table*

Description

Method to summarize the UMS of RIdata and RIcand and checks of the UMs overlap

Usage

```
createVerificationTab(RIperc, marginsRIdata, marginsRIcand)
```

Arguments

`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))
`marginsRIdata`	(data.frame) Margins for the data-derived reference intervals
`marginsRIcand`	(data.frame) Margins for the candidate reference intervals

Value

`(list) containing whether the point estimate and margin overlap tests passed, and the verification table`

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>

`defineSearchRegions` *Helper function to define search regions for mu and sigma and to get the region around the main peak 'ab'*

Description

The function estimates the start search regions for mu and sigma for each lambda. Further it determines an appropriate region around the main peak 'ab' that is used for all lambdas.

Usage

```
defineSearchRegions(x, lambdaVec, roundingBase, abEst = NULL)
```

Arguments

`x`	(numeric) values specifying data points comprising pathological and non-pathological values
`lambdaVec`	(numeric) transformation parameter for inverse Box-Cox transformation
`roundingBase`	(numeric) describing the rounding base of the dataset
`abEst`	(numeric) vector with already estimated abSearchReg and abHist for second definition of search regions

Value

(list) with (abEst, search region for mu and sigma)

Author(s)

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

`estimateAB` *Helper function to find region around the main peak of a distribution*

Description

Helper function to find region around the main peak of a distribution

Usage

```
estimateAB(x)
```

Arguments

`x`	(numeric) vector of data points

Value

(list) with two numeric vectors with lower and upper bound of region around the main peak used for 1) defining the search regions and 2) estimating the histogram with overlapping bins

Author(s)

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

`findMainPeak` *Helper function to find the main peak of a distribution*

Description

The function uses a combination of the area under the curve between valleys and the peak height to detect the main peak.

Usage

```
findMainPeak(x, ab, mStart, withHeight = FALSE, prevPeak = NULL)
```

Arguments

`x`	(numeric) vector of data points
`ab`	(numeric) vector specifying the lower and higher truncation limit of density estimation
`mStart`	(integer) specifying the width of the smoothing kernel(s) used for density estimation
`withHeight`	(logical) specifying if only the area under the curve (FALSE) or a combination of AUC and peak height (TRUE) should be used to detect the main peak
`prevPeak`	(numeric) specifying the modEst of the previously estimated peak

Value

(list) with the two numeric values peakInd, modEst, and a density list

Author(s)

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

`findPeaksAndValleys` *Find the index of the peaks and valleys of the density estimation.*

Description

Find the index of the peaks and valleys of the density estimation.

Usage

```
findPeaksAndValleys(Dens)
```

Arguments

`Dens`	(list) with density estimation (x values, y values)

Value

(list) specifying the index of the peaks and valleys of the density estimation.

Author(s)

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

`findRI` *Function to estimate reference intervals for a single population*

Description

The function estimates the optimal parameters lambda, mu and sigma for a raw data set containing pathological and non-pathological values. The optimization is carried out via a multi-level grid search to minimize the cost function (negative log-likelihood with regularization) and to find a model that fits the distribution of the physiological values and thus separates pathological from non-pathological values.

Usage

```
findRI(
  Data = NULL,
  model = c("BoxCox", "modBoxCoxFast", "modBoxCox"),
  NBootstrap = 0,
  seed = 123,
  ...
)
```

Arguments

`Data`	(numeric) values specifying data points comprising pathological and non-pathological values
`model`	(character) specifying the applied model (can be either "BoxCox" (default), "modBoxCoxFast" or "modBoxCox"), option "modBoxCoxFast" and "modBoxCox" first runs the original optimization using the Box-Cox transformation, afterwards the modified Box-Cox transformation is utilized and an optimal shift is identified ('fast': only 1 iteration is carried out to find a shift)
`NBootstrap`	(integer) specifying the number of bootstrap repetitions
`seed`	(integer) specifying the seed used for bootstrapping
`...`	additional arguments to be passed to the method

Value

(object) of class "RWDRI" with parameters optimized

Author(s)

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

Examples

```
# first example

resRI <- findRI(Data = testcase1)
print(resRI)
plot(resRI, showPathol = FALSE)

# second example
resRI <- findRI(Data = testcase2)
print(resRI, RIperc = c(0.025, 0.5, 0.975))
plot(resRI, showPathol = FALSE)

# third example, with bootstrapping
resRI <- findRI(Data = testcase3, NBootstrap = 30, seed = 123)
print(resRI)
getRI(resRI, RIperc = c(0.025, 0.5, 0.975), CIprop = 0.95, pointEst ="fullDataEst")
getRI(resRI, RIperc = c(0.025, 0.5, 0.975), CIprop = 0.95, pointEst ="medianBS")
plot(resRI)
```

```
# forth example, without values and pathological distribution in plot function
resRI <- findRI(Data = testcase4)
print(resRI)
plot(resRI, showValue = FALSE, showPathol =FALSE)

# fifth example, with bootstrapping
resRI <- findRI(Data = testcase5, NBootstrap = 30)
plot(resRI,  RIperc = c(0.025, 0.5, 0.975), showPathol = FALSE, showCI = TRUE)
```

`findRoundingBase` *Estimate rounding base of the input data.*

Description

Estimate rounding base of the input data.

Usage

```
findRoundingBase(x)
```

Arguments

`x`	(numeric) vector of data points

Value

(numeric) with estimated rounding base (e.g. 0.001 when rounded to 3 digits)

Author(s)

Christopher Rank <`christopher.rank@roche.com`>, Tatjana Ammer <`tatjana.ammer@roche.com`>

`generateHistData` *Generate list with histogram data.*

Description

Generate list with histogram data.

Usage

```
generateHistData(x, ab, roundingBase)
```

Arguments

`x`	(numeric) vector of data points
`ab`	(numeric) vector of lower and higher limit embedding appropriate region with the main peak
`roundingBase`	(numeric) describing the rounding base of the dataset

Value

(list) with histogram data used in the calculation of cost.

Author(s)

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

`getEquivalenceLimits` *Calculate equivalence limits*

Description

Calculate equivalence limits

Usage

```
getEquivalenceLimits(
  RI,
  RIperc = c(0.025, 0.975),
  CIprop = 0.9,
  pCVA.exp = 0.5,
  with.bias = FALSE,
  n = NULL
)
```

Arguments

`RI`	(numeric) vector of length 2 representing the lower and upper limits of the reference interval
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))
`CIprop`	(numeric) specifying the width of the confidence interval used to determine Equivalence Limits (default 0.9)
`pCVA.exp`	(numeric) value greater than 0 representing the exponent for the calculation of pCV.A
`with.bias`	(logical) value indicating whether to consider bias according to Haekel 2015
`n`	(numeric) value representing the sample size for consideration of bias

Value

`(data.frame) containing the calculated equivalence limits`

Author(s)

Matthias Beck <`matthias.beck.mb1@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"),
  Scale = c("original", "transformed", "zScore"),
  UMprop = 0.9,
  ...
)
```

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 model from all bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0
`Scale`	(character) specifying if percentiles are calculated on the original scale ("or") or the transformed scale ("tr") or the z-Score scale ("z")
`UMprop`	(numeric) value specifying the central region for estimation of uncertainty margins
`...`	calcUCMargins argument (logical) disabling the calculation of uncertainty margins when set to FALSE, n argument (integer) specifying the theoretical sample size used for uncertainty margin calculation default (n = 120), asymmetryCorr argument (logical) disabling the asymmetry correction when set to FALSE

Value

`(data.frame) with columns for percentile, point estimate, bootstrap-based confidence intervals and unce`

Author(s)

Christopher Rank <`christopher.rank@roche.com`>, Tatjana Ammer <`tatjana.ammer@roche.com`>, Matthias Beck <`matthias.beck.mb1@roche.com`>

`getRIfromRWDRI` *Get Reference Interval from RWDRI*

Description

Helper function to get the reference interval from a RWDRI object. For internal use only.

Usage

```
getRIfromRWDRI(RWDRI, RIperc, pointEst)
```

Arguments

`RWDRI`	(RWDRI) The RWDRI object containing the reference interval information.
`RIperc`	(numeric) The percentiles to interpolate.
`pointEst`	(character) The point estimate to use ("fullDataEst" or "medianBS").

Value

(list) A list containing the interpolated reference interval and percentiles.

`getRIMargins` *Calculate uncertainty margins for a reference interval using the asymptotic method*

Description

Calculate uncertainty margins for a reference interval using the asymptotic method

Usage

```
getRIMargins(
  RI,
  RIperc = c(0.025, 0.975),
  UMprop = 0.9,
  lambda = 0,
  shift = 0,
  asymmetryCorr = FALSE,
  n = 120,
  verbose = TRUE
)
```

Arguments

`RI`	(numeric) vector of length >= 2 representing the lower and upper limits of the reference interval.
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default `c(0.025, 0.975)`)
`UMprop`	(numeric) value between 0 and 1 representing the confidence level for the uncertainty margins.
`lambda`	(numeric) value representing the power parameter for the Box-Cox transformation.
`shift`	(numeric) value representing the shift parameter for the Box-Cox transformation.
`asymmetryCorr`	(logical) value if the asymmetry correction shall be applied for extremely skewed distributions
`n`	(numeric) value representing the sample size for which the sampling uncertainty shall be taken into account
`verbose`	(logical) specifying if additional warning messages are printed

Value

`(data.frame) containing the calculated uncertainty margins for the reference interval.`

Author(s)

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

Examples

```
## Not run:
getRIMargins(RI = c(12, 65), lambda = 0, shift = 0)
getRIMargins(RI = c(12, 65), lambda = 1, shift = 0)
 the following examples should return NAs with and without asymmetry correction
 # Next examples should return NAs
getRIMargins(RI = c(78.6, 116.7), lambda = 0.30673426, shift = 120.898)
getRIMargins(RI = c(78.6, 116.7), lambda = 0.30673426, shift = 120.898, asymmetryCorr = FALSE)
```

```
## End(Not run)
```

`getRISimilarity` *Calculate similarity of two reference intervals*

Description

Calculate similarity of two reference intervals

Usage

```
getRISimilarity(
  RIdata,
  RIcand,
  RIperc = c(0.025, 0.975),
  pointEst = c("fullDataEst", "medianBS"),
  UMprop = 0.9,
  Overlap = c("OverlapMargins", "OverlapPointEst"),
  printResults = TRUE,
  verbose = TRUE,
  ...
)
```

Arguments

`RIdata`	specifying the RI of the local population: (1) (object) of class `RWDRI` or (2) (numeric) representation of reference limits
`RIcand`	specifying the RI that needs to be verified: (1) (object) of class `RWDRI` or (2) (numeric) representation of reference limits
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))
`pointEst`	(character) specifying the point estimate determination: (1) using the full dataset ("fullDataEst"), (2) calculating the median model from the bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0
`UMprop`	(numeric) specifying the confidence level for the uncertainty margins
`Overlap`	(character) specifying the overlap criteria for the verification process. Options are: (1) uncertainty margins overlap "OverlapMargins" and (2) point estimates must be within the uncertainty margins "OverlapPointEst"
`printResults`	(logical) specifying if the results are printed to the console
`verbose`	(logical) specifying if additional warning messages are printed
`...`	arguments to overwrite the default values of the Uncertainty Margin calculation

Value

`(data.frame) containing the similarity of the two reference intervals`

Author(s)

Matthias Beck `<matthias.beck.mb1@roche.com>`

Examples

```
## Not run:
example <- list(Mu = 3.41, Sigma = 0.504, Shift = 1, Lambda = 0.06,
  Method = "manual", roundingBase = NA)
class(example) <- "RWDRI"
test <- getRISimilarity(example, c(4, 55.5))
getRISimilarity(c(4, 55.5), c(6, 58))
getRISimilarity(c(4, 55.5), c(6, 58), UMprop = 0.95)

## End(Not run)
```

`getSumForPArea` *Helper function to calculate the amount of observed and estimated data points within specified regions around the peak.*

Description

The function helps to define the search region for P (fraction of non-pathological samples).

Usage

```
getSumForPArea(
  pLimitMin,
  pLimitMax,
  countsPred,
  HistData,
  lambda,
  mu,
  sigma,
  pCorr
)
```

Arguments

`pLimitMin`	(numeric) vector specifying the lower limits for the regions next to the peak
`pLimitMax`	(numeric) vector specifying the upper limits for the regions next to the peak
`countsPred`	(numeric) vector with the predicted counts
`HistData`	(list) with histogram data generated by function `generateHistData`
`lambda`	(numeric) transformation parameter for inverse Box-Cox transformation

`mu`	(numeric) parameter of the mean of non-pathological distribution
`sigma`	(numeric) parameter of the standard deviation of non-pathological distribution
`pCorr`	(numeric) correcting the cumulative probability of the truncated non-pathological distribution

Value

(list) with two numeric vectors specifying the amount of observed and estimated data points surrounding the peak

Author(s)

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

`getVerificationArgs` *Get the correct values for RI, RIperc, Lambda, and Shift for the verification*

Description

This function adapts the list of test arguments based on the margin type and additional test parameters

Usage

```
getVerificationArgs(
  RIdata,
  RIcand,
  RIperc,
  pointEst = c("fullDataEst", "medianBS"),
  verbose = TRUE
)
```

Arguments

`RIdata`	specifying the RI of the local population: (1) (object) of class RWDRI or (2) (numeric) representation of reference limits
`RIcand`	specifying the RI that needs to be verified: (1) (object) of class RWDRI or (2) (numeric) representation of reference limits
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))
`pointEst`	(character) specifying the point estimate determination: (1) using the full dataset ("fullDataEst"), (2) calculating the median model from the bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0
`verbose`	(logical) specifying if additional warning messages are printed

Value

(list) containing a list with the "RI", "Lambda", "Shift" parameter for RIdata and RIcand each

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>

`invBoxCox` *Inverse of the one-parameter Box-Cox transformation.*

Description

Inverse of the one-parameter Box-Cox transformation.

Usage

```
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`>

`optimizeGrid` *Helper function for grid search for mu and sigma.*

Description

Helper function for grid search for mu and sigma.

Usage

```
optimizeGrid(currentBestParam, paramUnique, iter, sigmLimit = TRUE)
```

Arguments

`currentBestParam`	(numeric) value specifying the current best value for this parameter
`paramUnique`	(numeric) vector of possible values for this parameter
`iter`	(integer) indicating the number of iteration, as in the first iteration the search region is larger than in the following iterations
`sigmLimit`	(logical) specifiying if parameter is sigma and thus minimum is 0

Value

(vector) specifying the new search region fo the parameter to be optimized

Author(s)

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

`perc_ci_asymptotic` *Function to approximate the sampling uncertainty of quantiles using the asymptotic method*

Description

Function to approximate the sampling uncertainty of quantiles using the asymptotic method

Usage

```
perc_ci_asymptotic(mu, sigma, n, prob, conf.level)
```

Arguments

`mu`	(numeric) mean of the distribution
`sigma`	(numeric) standard deviation of the distribution
`n`	(numeric) integer value indicating the sample size
`prob`	(numeric) quantile value(s) for which to calculate the confidence interval(s)
`conf.level`	(numeric) confidence level for the interval(s)

Value

(matrix) of confidence interval(s) of quantiles of the normal distribution

Author(s)

Florian Dufey <`florian.dufey@roche.com`>

References

Serfling RJ. Approximation theorems of mathematical statistics. NY: John Wiley & Sons; 1980:121 p.

Examples

```
## Not run:
perc_ci_asymptotic(mu = 0, sigma = 1, n = 120, prob = c(0.025, 0.975), conf.level = 0.9)

## End(Not run)
```

`plot.RWDRI` *Standard plot method for objects of class 'RWDRI'*

Description

Standard plot method for objects of class 'RWDRI'

Usage

```
## S3 method for class 'RWDRI'
plot(
  x,
  Scale = c("original", "transformed", "zScore"),
  RIperc = c(0.025, 0.975),
  Nhist = 60,
  showMargin = TRUE,
  showPathol = FALSE,
  scalePathol = TRUE,
  showBSModels = FALSE,
  showValue = TRUE,
  uncertaintyRegion = c("bootstrapCI", "uncertaintyMargin"),
  CIprop = 0.95,
  UMprop = 0.9,
  pointEst = c("fullDataEst", "medianBS"),
  colScheme = c("green", "blue"),
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  title = NULL,
  ...
)
```

Arguments

`x`	(object) of class 'RWDRI'
`Scale`	(character) specifying if percentiles are shown on the original scale ("or") or the transformed scale ("tr") or the z-Score scale ("z")
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))
`Nhist`	(integer) number of bins in the histogram (derived automatically if not set)
`showMargin`	(logical) specifying if the specified margins, i.e. confidence intervals or uncertainty margins, shall be shown
`showPathol`	(logical) specifying if the estimated pathological distribution shall be shown
`scalePathol`	(logical) specifying if the estimated pathological distribution shall be weighted with the ration of pathol/non-pathol
`showBSModels`	(logical) specifying if the estimated bootstrapping models shall be shown
`showValue`	(logical) specifying if the exact value of the estimated reference intervals shall be shown above the plot
`uncertaintyRegion`	(character) specifying the type of the uncertainty region around point estimates
`CIprop`	(numeric) value specifying the central region for estimation of confidence intervals
`UMprop`	(numeric) value specifying the central region for estimation of uncertainty margins
`pointEst`	(character) specifying the point estimate determination: (1) using the full dataset ("fullDataEst"), (2) calculating the median model from the bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0
`colScheme`	(character) specifying color scheme of the non-pathological distribution and reference interval; choices are "green" and "blue"
`xlim`	(numeric) vector specifying the limits in x-direction
`ylim`	(numeric) vector specifying the limits in y-direction
`xlab`	(character) specifying the x-axis label
`ylab`	(character) specifying the y-axis label
`title`	(character) specifying plot title
`...`	additional arguments passed forward to other functions

Value

`The applied plot limits in x-direction (xlim) are returned.`

Author(s)

Christopher Rank <christopher.rank@roche.com>, Tatjana Ammer <tatjana.ammer@roche.com>

`plotRIVerification` *Plot method for RI verification*

Description

Plot method for RI verification

Usage

```
plotRIVerification(
  mar1,
  mar2,
  marginOverlap,
  Scale = c("original", "transformed", "splitXAxis"),
  lambda = 0,
  xlab = NULL,
  title = NULL,
  ...
)
```

Arguments

`mar1`	(data.frame) with output of function getRIMargins()
`mar2`	(data.frame) with output of function getRIMargins()
`marginOverlap`	(character) vector length of mar1; specifying if and how the margins overlap; options are: "PEOverlap", "MarsOverlap", "noOverlap"
`Scale`	(character) specifying if percentiles are shown on the original scale ("original") or the transformed scale ("transformed") or a split view of the x-axis ("splitX-Axis")
`lambda`	(numeric) specifying the power parameter (skewness) of the assumed distribution
`xlab`	(character) specifying the x-axis label
`title`	(character) specifying plot title
`...`	(additional arguments) candLabel (character) specifying the label for the candidate RI; dataLabel (character) specifying the label for the data-derived RI

Value

`(NULL) Instead, a plot is generated.`

Author(s)

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

`pnormApprox` *Approximate calculation of CDF of normal distribution.*

Description

Approximate calculation of CDF of normal distribution.

Usage

```
pnormApprox(q, pNormVal, mean = 0, oneOverSd = 1, oneOverH = 10)
```

Arguments

`q`	(numeric) vector of quantiles of data points
`pNormVal`	(numeric) vector of lookup table for pNorm
`mean`	(numeric) vector of mean values
`oneOverSd`	(numeric) reciprocal vector of sd values
`oneOverH`	(numeric) defining the precision of the approximation

Value

(numeric) vector of approximate CDFs of normal distribution.

Author(s)

Christopher Rank <`christopher.rank@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),
  uncertaintyRegion = c("bootstrapCI", "uncertaintyMargin"),
  CIprop = 0.95,
  UMprop = 0.9,
  pointEst = c("fullDataEst", "medianBS"),
  ...
)
```

Arguments

`x`	(object) of class 'RWDRI'
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval
`uncertaintyRegion`	(character) specifying the type of the uncertainty region around point estimates
`CIprop`	(numeric) value specifying the central region for estimation of confidence intervals
`UMprop`	(numeric) value specifying the central region for estimation of uncertainty margins
`pointEst`	(character) specifying the point estimate determination: (1) using the full dataset ("fullDataEst"), (2) calculating the median model from all bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0
`...`	additional arguments passed forward to other functions.

Value

`No return value. Instead, a summary is printed.`

Author(s)

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

Examples

```
## Not run:
x <- refineR::findRI(refineR::testcase1)
print(x, uncertaintyRegion = "bootstrapCI")
print(x, uncertaintyRegion = "uncertaintyMargin")

## End(Not run)
```

`printDataFractionWithinOutsideRI`

Print Data Fractions Within and Outside Reference Intervals

Description

This function prints a table showing the fractions of data points that are below, within, and above specified reference intervals. The function handles both lower and upper limits, or just one of them, based on the input.

Usage

```
printDataFractionWithinOutsideRI(tab, RIperc, data)
```

Arguments

`tab`	(dataframe) must contain columns `Percentile`, `RICandPointEst`, and `RIDataPointEst`.
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))
`data`	(numeric) vector of unknown length, possibly containing NAs

Value

`(NULL) Prints the verification table to the console`

Author(s)

Matthias Beck `<matthias.beck.mb1@roche.com>`

`printSimilarityTable` *Helper function to print the results of getRISimilarity*

Description

This function prints a formatted similarity table showing reference intervals and maximum sample sizes.

Usage

```
printSimilarityTable(SimilarityTab, RIperc)
```

Arguments

`SimilarityTab`	(data.frame) containing the similarity table. Must include: • Percentile: The percentile values. • RIdata: Reference interval data points. • RIcand: Reference interval candidate points. • max_sample_size: Maximum sample sizes. • s_value: S-values indicating the level of similarity.
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval

Value

`(NULL) Prints the similarity table to the console.`

`printTableHeader` *Helper function to print Table Header*

Description

Helper function to print Table Header

Usage

```
printTableHeader(headers, col_widths)
```

Arguments

`headers`	(character) vector of column headers
`col_widths`	(numeric) vector of column widths

Value

`(NULL) Prints the header to the console.`

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>

`printTableRows` *Helper function to print Table Rows*

Description

Helper function to print Table Rows

Usage

```
printTableRows(rows, col_widths)
```

Arguments

`rows`	(list) of lists containing row data.
`col_widths`	(numeric) vector of column widths.

Value

`(NULL) Prints the rows to the console.`

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>

`printVerificationTable` *Print Verification Table*

Description

This function prints a formatted verification table for reference intervals

Usage

```
printVerificationTable(verificationTab, RIperc)
```

Arguments

`verificationTab`	(data.frame) containing the verification table. Must include the following columns: • Percentile: The percentile values. • RIDataPointEst: Point estimates for RI Data. • RIDataMarginLow: Lower margins for RI Data. • RIDataMarginHigh: Upper margins for RI Data. • RITestPointEst: Point estimates for RI Cand. • RITestMarginLow: Lower margins for RI Cand. • RITestMarginHigh: Upper margins for RI Cand. • OverlapPointEst: Logical indicating if point estimates overlap. • OverlapMargins: Logical indicating if margins overlap.
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))

Value

None. Prints the verification table to the console.

Examples

```
## Not run:
df <- data.frame(
Percentile = c(0.025, 0.975),
RICandPointEst = c(12, 65),
RICandMarginLow = c(10.9013, 60.2780),
RICandMarginHigh = c(13.86800, 72.64152),
RIDataPointEst = c(13, 69),
RIDataMarginLow = c(11.60130, 63.77848),
RIDataMarginHigh = c(14.39861, 74.22152),
OverlapPointEst = c(TRUE, TRUE),
OverlapMargins = c(TRUE, TRUE)
)
```

```
RIperc <- c(0.025, 0.975)
printVerificationTable(df, RIperc)

## End(Not run)
```

`testcase1` *Simulated Testcase 1.*

Description

This dataset consists of N = 10,000 simulated measurements with 80\ Ground Truth for reference intervals (2.5\

Usage

```
testcase1
```

Format

Numeric vector with data points.

`testcase2` *Simulated Testcase 2.*

Description

This dataset consists of N = 50,000 simulated measurements with 60\ Ground Truth for reference intervals (2.5\

Usage

```
testcase2
```

Format

Numeric vector with data points.

`testcase3` *Simulated Testcase 3.*

Description

This dataset consists of N = 75,000 simulated measurements with 96\ Ground Truth for reference intervals (2.5\

Usage

```
testcase3
```

Format

Numeric vector with data points.

`testcase4` *Simulated Testcase 4.*

Description

This dataset consists of N = 100,000 simulated measurements with 90\ Ground Truth for reference intervals (2.5\

Usage

```
testcase4
```

Format

Numeric vector with data points.

`testcase5` *Simulated Testcase 5.*

Description

This dataset consists of N = 250,000 simulated measurements with 80\ Ground Truth for reference intervals (2.5\

Usage

```
testcase5
```

Format

Numeric vector with data points.

`testParam` *Helper function to find optimal parameters lambda, mu and sigma.*

Description

Helper function to find optimal parameters lambda, mu and sigma.

Usage

```
testParam(
  lambdaVec,
  bestParam,
  Data,
  HistData,
  startValues,
  NIter,
  alpha = 0.01,
  alphaMcb = 0.1
)
```

Arguments

`lambdaVec`	(numeric) transformation parameter for inverse Box-Cox transformation
`bestParam`	(numeric) vector containing best guess for lambda, mu, sigma, P, cost
`Data`	(numeric) values specifying percentiles or data points comprising pathological and non-pathological values
`HistData`	(list) with histogram data
`startValues`	(list) with start search regions for mu and sigma
`NIter`	(integer) specifying the number of iterations for optimized grid-search
`alpha`	(numeric) specifying the confidence region used for selection of histogram bins in cost calculation
`alphaMcb`	(numeric) specifying the confidence level defining the maximal allowed counts below the asymmetric confidence region

Value

(numeric) vector with best parameters for lambda, mu, sigma, P, cost.

Author(s)

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

`verifyRI` *Verify Reference Intervals*

Description

This function verifies the reference intervals based on the provided data.

Usage

```
verifyRI(
  RIdata,
  RIcand,
  RIperc = c(0.025, 0.975),
  UMprop = 0.9,
  pointEst = c("fullDataEst", "medianBS"),
  printResults = TRUE,
  generatePlot = TRUE,
  Scale = c("original", "transformed", "splitXAxis"),
  xlab = NULL,
  title = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

`RIdata`	specifying the RI of the local population: (1) (object) of class `RWDRI` or (2) (numeric) representation of reference limits
`RIcand`	specifying the RI that needs to be verified: (1) (object) of class `RWDRI` or (2) (numeric) representation of reference limits
`RIperc`	(numeric) value specifying the percentiles, which define the reference interval (default c(0.025, 0.975))
`UMprop`	(numeric) specifying the width of the confidence interval approximated by the uncertainty margins (default 0.9)
`pointEst`	(character) specifying the point estimate determination: (1) using the full dataset ("fullDataEst"), (2) calculating the median model from the bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0
`printResults`	(logical) if TRUE, results are printed in the console.
`generatePlot`	(logical) if TRUE, a plot is generated.
`Scale`	(character) specifying the scale of the plot: (1) "original", (2) "transformed", (3) "splitXAxis"
`xlab`	(character) optional x-axis label for the plot
`title`	(character) optional title for the plot
`verbose`	(logical) specifying if additional warning messages are printed
`...`	(list) specifying non-default parameters for the calculation of the margins

Value

`(list) containing the verification results`

Author(s)

Matthias Beck <`matthias.beck.mb1@roche.com`>

Examples

```
## Not run:

# standard usecase:
# comparison of the RI estimated from the local population with a numeric candidate
# RI from, e.g., literature
RIdata <- findRI(testcase1)
RIcand <- c(10, 20)
verifyRI(RIdata, RIcand, RIperc = c(0.025, 0.975))

# Test if two refineR models are equivalent with stricter criteria, i.e., larger n
model1 <- list(Mu = 20, Sigma = 11, Shift = 8, Lambda = 1,
Method = "manual", roundingBase = NA)
model2 <- list(Mu = 3.41, Sigma = 0.504, Shift = 1, Lambda = 0.06,
Method = "manual", roundingBase = NA)
class(model1) <- class(model2) <- "RWDRI"
verifyRI(RIdata = model1, RIcand = model2, UMprop = 0.95, n = 1e4)

# verify two numeric RIs
verifyRI(RIdata = c(2.4, 28), RIcand = c(3.1, 29.75))

## End(Not run)
```

Index