Package: ratios (via r-universe)

August 21, 2024

Type Package

Title Calculating Ratios Between Two Data Sets and Correction for Adhering Particles on Plants

Version 1.2.0

Author Solveig Pospiech [aut, cre], Wiebke Fahlbusch [ctb]

Maintainer Solveig Pospiech <s.pospiech@gmx.de>

Description Calculation of ratios between two data sets containing environmental data like element concentrations by different methods. Additionally plant element concentrations can be corrected for adhering particles (soil, airborne dust).

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Depends R (>= 2.10)

Imports stringr, data.table

NeedsCompilation no

Date/Publication 2018-02-18 18:39:37 UTC

Repository https://solmonta.r-universe.dev

RemoteUrl https://github.com/cran/ratios

RemoteRef HEAD

RemoteSha c782fd047a71f63dbea036e5614e28ec21dc96e0

Contents

---

ratios-package   *Ratios of environmental data*

Description

The package provides functions for calculating ratios between two data sets containing environmental data like concentration of elements. Ratios can be calculated either by method "simple", "clr" or "alr". Additionally for plants the amount of adhering particles on plants can be estimated and the element concentration corrected by subtraction.

Details

Ratios:

Calculating ratios is at the first glance a simple operation but it becomes quickly more complex if the two data sets don't have corresponding rows or columns. A set of functions helps to faster and saver calculate ratios: If for a data set DT1 ratios to a second data set DT2 should be calculated the function preparationDT2 creates a 'new DT2' with equal number of rows and corresponding columns to DT1 with entries and mean of entries from DT2. Errors are calculated by the function relError_dataset as well for each data set as for the ratios, too. The function ratio.DT1_DT2 provides methods for six different types of ratios: 1. simple ratios 2. log ratios 3. ar ratios 4. alr ratios 5. cr ratios 6. clr ratios

clr and alr ratios are developed from the clr (center logarithmic transformation) and alr (additive logarithmic transformation) concept introduced by Aitchison in 1986 for compositional data, which are data constrained by a constant sum like concentrations of elements. Hence especially these two methods might be of interest if DT1 and DT2 contain compositional data. This is probably the case for most environmental data. The methods 'ar' and 'cr' are the same as 'alr' and 'clr', but without the logarithm.

Correction for Adhering Particles

Exact and reproducible analysis of element concentrations in plant tissue is the basis for many research fields such as environmental, health, phytomining, agricultural or provenance studies. Unfortunately plant samples collected in the field will always contain particles on their tissue surfaces such as airborne dust or soil particles. If not removed these particles may induce a bias to the element concentrations measured in plant samples. The influence of adhering particles on element concentration in plants is negligible for elements which have a much higher concentration in the plant tissues compared to the adhering material. This is the case for most main or minor nutrient elements such as P, K, Ca, Mg, S, Mn, B, Mo, Zn or Cu. But elements with typically very low concentrations in plant tissue such as Al, Co, Fe, Li, Ni, Ti, Sc, Zr, or REEs, may show significantly altered concentrations measured in the plants due to adhering particles. Mitchell (1960) proposed that elements with concentration ratios of soil to plant above 100 might show biased concentrations in the measured samples.

Reducing the impact of adhering particles on trace element concentration in plants is crucial in order to be able to compare elemental composition of plants, e.g. between sampling periods or slightly different sampling methods or for biomonitoring studies. It is also important in studies for plant nutrition to calculate the real uptake of an element by a plant, e.g. for phytoremediation/phytomining or in studies on the trace elements Co, Ni, Mn and Mo.

Based on the model that the analyzed plant material is a mixture of plant tissue and a very minor amount of adhering particles we developed a general methods to calculate a correction term for adhering material.

The function Correction.AdheringParticles provides three different methods to calculate the influence of adhering particles in order to obtain the element concentrations in plants resulting only from uptake.

For further reading and details please refer to the publication: Pospiech, S., Fahlbusch, W., Sauer, B., Pasold, T., & Ruppert, H. (2017). Alteration of trace element concentrations in plants by adhering particles–Methods of correction. Chemosphere, 182, 501-508.

---

check_readline   *check_readline*

Description

The function checks if a given character string, e.g. value from readline, matches one element of on a given character vector. Without match it keeps asking for a character string by readline until 'q' for quit or a matching string is provided. This is often used to avoid crashing of functions due to wrong input in the options.

Usage

```
check_readline(x, myletters)
```

Arguments

x   character vector to be checked, e.g. value from readline

myletters   character vector of entries which should be allowed, e.g. c("yes", "no")

Details

Usage is e.g. yesno = check_readline(yesno, c("y", "n")) Now the function will make sure that the variable *x* consists either of "y" or of "n".

Value

character vector with one of the values given in myletters

Author(s)

Solveig Pospiech

See Also

Other sub functions: `relError_dataset`, `select.VarsElements`

Examples

```
possibleEntries = c("today", "yesterday")
myEntry = "today"
# or try another entry which is different from "today":
# myEntry = readline("Enter any word (without quotes):    ")
y = check_readline(x = myEntry, myletters = possibleEntries)
```

Correction.AdheringParticles

Correction.AdheringParticles

Description

Suppose element data of one data set (DT1) are biased because the concentrations are the result of a mixture of two substances, of which one substance are the element concentrations of DT2. In order to correct DT1 to $DT_{corrected}$ a fraction of DT2 has to be subtracted from DT1. The basic equation for the correction is:

$$DT_{corrected} = \frac{DT1 - x * DT2}{1 - x}$$

whereof *x* is the amount of DT2 to be subtracted.

The function is written for the case that *x* is unknown. To calculate *x* the condition is that in $DT_{corrected}$ at least one element concentration is zero or known. Suppose $vars_i$ has a very low concentration, close to zero, in $DT_{corrected}$: $DT_{corrected}[vars_i] = 0$, then:

$$x = \frac{DT1[vars_i]}{DT2[vars_i]}$$

The function was developed for the use to correct plant concentrations for adhering particles: Exact and reproducible analysis of element concentrations in plant tissue is the basis for many research fields such as environmental, health, phytomining, agricultural or provenance studies. Unfortunately plant samples collected in the field will always contain particles on their tissue surfaces such as airborne dust or soil particles. If not removed these particles may induce a bias to the element concentrations measured in plant samples.

For full description of the calculations and the background of correction plants for adhering particles please refer to:

Pospiech, S., Fahlbusch, W., Sauer, B., Pasold, T., & Ruppert, H. (2017). Alteration of trace element concentrations in plants by adhering particles–Methods of correction. *Chemosphere, 182, 501-508*. and the section Details.

Usage

```
Correction.AdheringParticles(DT1, DT2 = NULL, vars = NULL,
  vars_ignore = c("As", "Se", "Sn", "V", "Be", "Ge", "Pt"), method, element,
  id.vars, group1.vars, group2.vars, var_subgroup, offset = 0,
  use_only_DT2 = TRUE, DT2_replace = NULL, Errors = TRUE,
  return_as_list = TRUE, negative_values = FALSE,
  set_statistical_0 = FALSE, Error_method = "gauss", STD_DT1 = STD_Plant,
  STD_DT2 = STD_Soil, minNr_DT1 = 100, minNr_DT2 = 100)
```

Arguments

DT1	data.frame or data.table, samples in rows and variables in columns
DT2	data.frame or data.table, samples in rows and variables in columns.
vars	optional, character vector of column names of DT1 and DT2, default is function [select.VarsElements](). Please make sure the columns given in vars are of class numeric.
vars_ignore	character vector of column names, only for 'method 3'. These variables are ignored for calculating the median of amount of DT2 (x) in 'method 3'. Please note: the functions returns corrected values for these columns because they are only ignored for calculating the median of x. Default is "As", "Se", "Sn", "V", "Be", "Ge" and "Pt". Please see Details for further explanation.
method	characters (no character vector!, please give m3 instead of "m3") denoting the method. Options are *m1*, *m2* and *m3* and *subtr*. Default is m3. Please see details.
element	string, only for method 1. Denotes the column with which amount of DT2 (x) is to be calculated.
id.vars	column with unique (!) entries for each row. Class can be integer (corresponding row numbers) or character (e.g. sample IDs). If missing, all columns but vars will be assigned to it. Please note: Function is faster and more stable if id.vars is provided.
group1.vars	character vector, column name(s) for subsetting DT1 and DT2
group2.vars	optional, column name for subsetting DT1 and DT2 if some entries in group1.vars are empty.
var_subgroup	optional, character vector of one column name of DT1. This option affects the only the error calculation, hence it is ignored if Errors is set to FALSE. If provided, DT1 is split into subsets by group1.vars *and* 'var_subgroup' and the error will calculated for each of these subset. Please read in the Details for further information.
offset	numeric, default is 0. The offset diminishes the subtracted amount of DT2 x: x = x - offset. If used with m2 all concentrations will stay > 0. Reasonable offset is e.g. offset = 0.0001
use_only_DT2	logical, default is FALSE. If there are not enough DT2 data of the location should the DT2s of the region be used? If the use_only_DT2 is set to FALSE then the Upper Crust is used for the correction.

DT2_replace	optional, if a DT1 sample does not have DT2 data of the corresponding location with this option you can define which data you would like to use as DT2. Default is the build-in data set UpperCrust (geochemical composition of the earth's upper crust). If you would like to have something else, please provide a named vector/ one-row data.table with values used instead of DT2.
Errors	logical, should absolute errors get calculated appended to the list - output? Default is FALSE. If Errors are set to TRUE it overrides the option `return_as_list` and always returns a list.
return_as_list	logical, should the result get returned as list? Default is FALSE.
negative_values	logical, should negative values be returned? If set to FALSE negative values are set to 0. Default is FALSE.
set_statistical_0	logical, only for method 3. Should all values of the variables contributing to the median of *x* be set to 0? Default is FALSE.
Error_method	method with which the error should be calculated. At the moment you can choose between "gauss" (default) and "biggest". See Details for explanation.
STD_DT1	optional, data.frame or data.table object for calculating errors for DT1, e.g. the standards. Please see Details. If left empty a default of 5.2% relative error is used.
STD_DT2	optional, data.frame or data.table object for calculating errors for DT2, e.g. the standards. Please see Details. If left empty a default of 5.2% relative error is used.
minNr_DT1	minimum numbers of samples/observations in DT1 for calculating a relative error of observations. If the number of observations of DT1 is smaller than `minNr_DT1` the error is calculated via the data set `STD_DT1`. Default is 50.
minNr_DT2	minimum numbers of samples/observations in DT2 for calculating a relative error of observations. If the number of observations of DT1 is smaller than `minNr_DT2` the error is calculated via the data set `STD_DT2`. Default is 50.

Details

The main option of this function is the `method` which determines how the amount of DT2 to be subtracted, the *x*, is going to be calculated. There are four options:

- Method 1: calculate *x* via a fixed element
- Method 2: calculate *x* via the element with the smallest ratio between DT1[vars] and DT2[vars]
- Method 3: calculate *x* via the median of several, very small ratios between DT1[vars] and DT2[vars]
- Method subtr: calculate the concentrations for $x * DT2[vars]$

To Method 1: For example using Ti as element $DT_{corrected}$ is calculated with $x = DT1[Ti]/DT2[Ti]$. Typical elements for the option element are e.g. Ti, Al, Zr, Sc, ... This will eventually lead to negative concentrations for some elements.

To Method 2: This method subtracts the smallest possible content of DT2 from DT1 (smallest *x*). For each row/sample the element with the smallest *x* of all ratios $x = DT1[vars]/DT2[vars]$ of

each sample is taken as *element*, hence every sample is corrected based on a different *element*. With this method there are no negative concentrations.

To Method 3: In order to reduce the uncertainty of the content of DT2 in DT1 (x) based on only one element as in method 1 and 2 an average of several x of elements can be calculated. With Δx being the absolute error of x the median is calculated by all x of elements which values $x - \Delta x$ are smaller than $x_{smallest} + \Delta x_{smallest}$. The value of the median $\bar{x}$ is then used as x. This will eventually lead to negative concentrations for some elements. Because statistically the x of all elements, which error overlaps the error of the element with smallest x, are indistinguishable we suggest to set all elements contributing to $\bar{x}$ to zero, because these small values should not be interpreted: Set option `set_statistical_0` to TRUE.

It is advisable to exclude elements with a huge error margin in the option `vars_ignore` because they could severely increase the median $\bar{x}$ by "opening" the window of error-ranges for many elements with significantly higher ratios. This could lead to an unnatural high median $\bar{x}$ resulting into an overcorrection.

If option `id.vars` is provided the functions prints the 'group1.vars' and 'id.vars' of the sample.

For examples and more information please refer to: Pospiech, S., Fahlbusch, W., Sauer, B., Pasold, T., & Ruppert, H. (2017). Alteration of trace element concentrations in plants by adhering particles–Methods of correction. *Chemosphere, 182, 501-508.*

Value

data.frame (or data.table if DT1 is data.table) according to `method`.

Author(s)

Solveig Pospiech

See Also

Other ratio functions: `preparationDT2`, `ratioDT`, `ratio_append_smallest`

---

preparationDT2 *preparationDT2*

Description

The function creates a data frame 'new DT2' from the variables `vars` of the data set DT2 with corresponding rows to the data set DT1, hence 'new DT2' and DT1 have the same number of rows. The aim is to generate corresponding rows for two data sets with differing dimensions and even differing number of rows for each group of rows. For example if for one row i in DT1 there are 3 corresponding rows (j,k,l) in DT2 the function calculates for the 'new DT2' for each variable of `vars` an average over the rows j,k and l of DT2, generating only one row corresponding to the row i in DT1. If on the other hand for row y in DT2 there are 4 corresponding rows in DT1 the 'new DT2' will contain four times the row y of DT2 matching the four rows of DT1. The column `group1.vars` (and optional `group2.vars`) determines which rows of DT1 and DT2 are corresponding, so `group1.vars` in DT1 is the look-up table for creating the 'new DT2'. Generally DT1 and DT2 have to have the columns in common which are given in `group1.vars`, `group2.vars` and `vars`.

Usage

```
preparationDT2(DT1, DT2, vars = NULL, group1.vars, group2.vars = NULL,
  Errors = FALSE, use_only_DT2 = FALSE, DT2_replace = NULL, minNr = 7,
  STD = NULL, return_as_list = FALSE)
```

Arguments

DT1	data.frame or data.table, samples in rows and variables in columns
DT2	data.frame or data.table, samples in rows and variables in columns.
vars	optional, character vector of column names of DT1 and DT2, default is function `select.VarsElements`. Please make sure the columns given in `vars` are of class numeric.
group1.vars	character vector, column name(s) for subsetting DT1 and DT2
group2.vars	optional, column name for subsetting DT1 and DT2 if some entries in `group1.vars` are empty.
Errors	logical, should absolute errors get calculated appended to the list - output? Default is FALSE. If Errors are set to TRUE it overrides the option `return_as_list` and always returns a list.
use_only_DT2	logical, default is FALSE. If there are not enough DT2 data of the location should the DT2s of the region be used? If the `use_only_DT2` is set to FALSE then the Upper Crust is used for the correction.
DT2_replace	mandatory if `use_only_DT2` is set to FALSE, serves as substitute for DT2 where DT2 has no corresponding rows to DT1. A named vector or one-row data.table/ data.frame with the all `vars` present. A column for `group1.vars` is not necessary.
minNr	minimum numbers of samples/observations for calculating a relative error of observations. If the number of samples of `DT2` is smaller than `minNr` the error is calculated via the data set STD.
STD	data set for calculating the relative errors if in DT2 there are less rows per group than `minNr`. This replacement data set could for e.g. consist of reference standards with repeated measurement for each standard.
return_as_list	logical, should the result get returned as list? Default is FALSE.

Details

The data set 'new DT2' is generated according to following rules: If there is more than one row in DT2 with the same entry for `group1.vars` for each column in `vars` an average (mean) of these rows of DT2 is calculated. After this operation there is only one row for each entry value of `group1.vars`. Each row of this averaged DT2 is replicated n times, with n being the number of rows of the subset of DT1 with the corresponding value in `group1.vars`. If there are values in column `group1.vars` in DT1 which are not in DT2 and if option `use_only_DT2` is set to TRUE empty rows are generated. If option `use_only_DT2` is set to FALSE, data from 'DT2_replace' are taken as substitute for DT2 to fill these empty rows. The default 'DT2_replace' are element concentrations from the UpperCrust (Rudnick, R. L., & Gao, S. 2003. Composition of the continental crust. *Treatise on geochemistry, 3*, 659.)

Value

data.frame, data.table or a list, controlled by option return_as_list. If Errors is set to TRUE return_as_list is ignored and return value is always a list. The list contains one element if Errors is set to FALSE and two elements if Errors is TRUE: [[1]] is data.table or data.frame of corresponding DT2s, [[2]] data.table or data.frame of absolute errors of corresponding DT2s.

Author(s)

Solveig Pospiech

See Also

Other ratio functions: `Correction.AdheringParticles`, `ratioDT`, `ratio_append_smallest`

---

ratioDT   *ratioDT*

Description

The function calculates ratios of corresponding variables and corresponding rows between two data sets, DT1 and DT2. The result is a data set with the same dimensions as DT1. The variables can be specified by vars, without specification the subfunction `select.VarsElements` matches column names with element abbreviations. Which row of DT1 corresponds to which row in DT2 has to be specified by the variable(s) group1.vars (and optional group2.vars). If DT2 has different number of rows than DT1 a 'new DT2' with equal dimensions to DT1 is prepared by the function `preparationDT2`. At the moment there are three different options for calculating the ratios:

- "simple"
- "log"
- "ar"
- "alr"
- "cr"
- "clr"

For more details please refer to `preparationDT2` and section Details.

Usage

```
ratioDT(DT1, DT2, vars = NULL, group1.vars, group2.vars = NULL,
  ratio_type = "simple", vars.ref, id.vars, Errors = FALSE,
  Error_method = "gauss", var_subgroup = NULL, use_only_DT2 = FALSE,
  DT2_replace = NULL, STD_DT1, STD_DT2, minNr_DT1 = 50, minNr_DT2 = 50,
  return_all = FALSE, return_as_list = FALSE)
```

Arguments

DT1	data.frame or data.table, samples in rows and variables in columns
DT2	data.frame or data.table, samples in rows and variables in columns.
vars	optional, character vector of column names of DT1 and DT2, default is function [select.VarsElements](). Please make sure the columns given in vars are of class numeric.
group1.vars	character vector, column name(s) for subsetting DT1 and DT2
group2.vars	optional, column name for subsetting DT1 and DT2 if some entries in group1.vars are empty.
ratio_type	character vector of "simple", "log", "ar", "alr", "cr" and "clr". Please refer to details for explanations.
vars.ref	reference variable, one out of vars. Only for ratio_type "ar" or "alr".
id.vars	column with unique (!) entries for each row. Class can be integer (corresponding row numbers) or character (e.g. sample IDs). If missing, all columns but vars will be assigned to it. Please note: Function is faster and more stable if id.vars is provided.
Errors	logical, should absolute errors get calculated appended to the list - output? Default is FALSE. If Errors are set to TRUE it overrides the option return_as_list and always returns a list.
Error_method	method with which the error should be calculated. At the moment you can choose between "gauss" (default) and "biggest". See Details for explanation.
var_subgroup	optional, character vector of one column name of DT1. This option affects the only the error calculation, hence it is ignored if Errors is set to FALSE. If provided, DT1 is split into subsets by group1.vars *and* 'var_subgroup' and the error will calculated for each of these subset. Please read in the Details for further information.
use_only_DT2	logical, default is FALSE. If there are not enough DT2 data of the location should the DT2s of the region be used? If the use_only_DT2 is set to FALSE then the Upper Crust is used for the correction.
DT2_replace	mandatory if use_only_DT2 is set to FALSE, serves as substitute for DT2 where DT2 has no corresponding rows to DT1. A named vector or one-row data.table/ data.frame with the all vars present. A column for group1.vars is not necessary.
STD_DT1	optional, data.frame or data.table object for calculating errors for DT1, e.g. the standards. Please see Details. If left empty a default of 5.2% relative error is used.
STD_DT2	optional, data.frame or data.table object for calculating errors for DT2, e.g. the standards. Please see Details. If left empty a default of 5.2% relative error is used.
minNr_DT1	minimum numbers of samples/observations in DT1 for calculating a relative error of observations. If the number of observations of DT1 is smaller than minNr_DT1 the error is calculated via the data set STD_DT1. Default is 50.

minNr_DT2	minimum numbers of samples/observations in DT2 for calculating a relative error of observations. If the number of observations of DT1 is smaller than minNr_DT2 the error is calculated via the data set STD_DT2. Default is 50.
return_all	logical, should *all* used data sets be returned as a list? Default is FALSE. If set to TRUE the list contains DT1, DT2, vars, ratios, and optional additional ratios_error, DT1_error and DT2_error.
return_as_list	logical, should the result get returned as list? Default is FALSE. If set to FALSE and Errors is set to TRUE a column type_of_data is appended. This option is ignored if option 'return_all' is set to TRUE.

Details

To calculate the ratios the functions internally calls preparationDT2 to create a data set 'new DT2' from the variables vars of DT2, which has equal number of rows to DT1. Then the division is done by the now corresponding data sets by the method given in 'ratio_type'.

The method "simple" is a simple division between DT1 and DT2:

$$\frac{DT1[vars]}{DT2[vars]}$$

The method "log" is the logarithm of the simple ratio:

$$ln\left(\frac{DT1[vars]}{DT2[vars]}\right)$$

The methods "ar" and "alr" normalize all ratios to one reference column: ar:

$$\frac{DT1[vars_i]}{DT2[vars_i]} * \frac{DT2[vars_n]}{DT1[vars_n]}_{i=1,\dots,n,\dots,D}$$

alr:

$$ln\left(\frac{DT1[vars_i]}{DT2[vars_i]} * \frac{DT2[vars_n]}{DT1[vars_n]}\right)_{i=1,\dots,n,\dots,D}$$

The methods "cr" and "clr" normalize all ratios to the geometric mean of all columns included by vars: "cr" is calculated by:

$$\frac{DT1[vars_i]}{DT2[vars_i]} * \frac{g(x)^{DT2[vars]}}{g(x)^{DT1[vars]}}_{i=1,\dots,D}$$

whereof the function g(x) stands for:

$$g(x) = \sqrt[D]{DT[vars_1] \cdot DT[vars_2] \cdots DT[vars_D]}$$

and "clr" is calculated by:

$$ln\left(\frac{DT1[vars_i]}{DT2[vars_i]} * \frac{g(x)^{DT2[vars]}}{g(x)^{DT1[vars]}}\right)_{i=1,\dots,D}$$

The methods "clr" and "alr" should be considered if the data contain so called *compositional data* as defined by Aitchison, J. (1986): "The statistical analysis of compositional data". They names correspond to the names used in the package compositions by K. Gerald van den Boogaart, Raimon Tolosana and Matevz Bren.

Calculating the absolute error for the ratios requires calculating the absolute errors of DT1 and DT2, too. For calculating the errors of DT1 and DT2 the function [relError_dataset](#) is used. Accordingly the options for STD_DT1 and STD_DT2 are passed to the option STD in relError_dataset. If STD_DT1 and/or STD_DT2 are left empty the default of 5.2% relative error is used. Also the options minNr_DT1 and minNr_DT2 are passed to the option minNr in relError_dataset.

The Error_method determines how the absolute error of the ratios is calculated. The error method "gauss" refers to the error propagation after Gauss:

$$\Delta x = \frac{\Delta DT1}{DT2} - DT1 * \frac{\Delta DT2}{DT2^2}$$

The error method "biggest" refers to the maximum error after Gauss:

$$\Delta x = \frac{\Delta DT1}{DT2} + DT1 * \frac{\Delta DT2}{DT2^2}$$

For example: If you have in DT1 plant samples with group1.vars = "Location" the error function would calculate the relative standard deviation for all plants of one location. But maybe you have very different plants in one location so setting var_subgroup = "Species" the error function will calculate the relative standard deviation for each plant species per location, if there are more species per location than given in minNr_DT1. Suppose DT2 are soil data with several samples per location. If group1.vars = "Location" than the function calls [preparationDT2](#) and calculates a mean for each location from the data set. The ratio from plant to soil and the absolute errors of the ratios is then calculated for each plant sample to a mean of soils from one location.

Value

The function returns either a data.table, data.frame or a list controlled by the option return_as_list. If return_as_list to FALSE a data.frame (or data.table if DT1 is of class data.table) is returned. If option Errors is set to TRUE ratios and error are combined into one object and a column type_of_data is appended with the entries *ratio* and *ratio_error* respectively. If return_as_list to TRUE the DT1-DT2-ratios are named in the list as "ratios" and, if Errors is set to TRUE the absolute errors of the ratios are saved in the list as "ratios_error". If 'return_all' is set to TRUE a list with the following entries will be returned:

[[1]] "DT1", [[2]] "DT2", [[3]] "vars", [[4]] "ratios" and if Errors is set to TRUE additionally [[5]] "ratios_error", [[6]] "DT1_error", [[7]] "DT2_error".

Author(s)

Solveig Pospiech

See Also

Other ratio functions: [Correction.AdheringParticles](#), [preparationDT2](#), [ratio_append_smallest](#)

---

ratio_append_smallest *ratio_append_smallest*

Description

The function appends for each row the smallest ratio DT1/DT2 in the column *ratio_smallest*. The name of the column which contained the smallest ratio is appended in the column *ratio_smallest_Elem*. This function is basically a sub-function for the function Correction.AdheringParticles.

Usage

```
ratio_append_smallest(Ratios, vars = NULL)
```

Arguments

Ratios   list, data.frame or data.table, which is the output after using the function ratioDT

vars   optional, character vector of column names of DT1 and DT2, default is function
                select.VarsElements. Please make sure the columns given in vars are of
                class numeric.

Value

list with [[1]] being the data set from the input with one column added containing the smallest ratio of all variables given in vars. If the input was a list with one element named "ratios_error" the returned list contains a second element [[2]] "ratios_error" also with the appended columns.

Author(s)

Solveig Pospiech

See Also

Other ratio functions: Correction.AdheringParticles, preparationDT2, ratioDT

---

relError_dataset *relError_dataset*

Description

The function calculates for each observation for every variable 'vars' in 'Data' the relative error by median absolute deviation (mad) and median (median):

$$\delta Data[vars_i] = \frac{mad(Data[vars_i], na.rm = T)}{median(Data[vars_i], na.rm = T)}$$

The observations (e.g. samples) are subset into groups by the column group1.vars. The relative error is calculated by 'Data' if there are more than 'minNr' entries for each subset of observations.

If there are less observations than 'minNr' for a group in 'Data' than the relative error will be calculated by a replacement data set 'STD', e.g. you could use a data set of standard reference samples measured at the same machine as your samples. If you would like to calculate the relative error of all observations in 'Data' set group1.vars to the column of your sample ID (column with unique entries) and set minNr = 1.

Usage

```
relError_dataset(Data, vars, group1.vars, group2.vars = NULL, minNr = 7,
  STD)
```

Arguments

Data	a data.frame or matrix with samples (observations) as rows.
vars	optional, character vector of variables of 'Data' for which the error should be calculated. If left empty the function select.VarsElements will try to find element abbreviations in the variables of 'Data' and 'STD' if STD is provided.
group1.vars	character vector of variables in 'Data' for splitting 'Data' into subsets. Error will be calculated for each subset.
group2.vars	optional, if a variable name of 'Data' is given here a second splitting by group1.vars + group2.vars into subsets is performed. If for grouping by group1.vars for one subset there are less entries than 'minNr' the function will look up in the second subset if there are enough entries (> minNr) in the group2 corresponding to group1. For example if group1.vars = "Month" then group2.vars = "Year" would fill up the gaps if in one month there had been less than 'minNr' observations.
minNr	minimum numbers of samples/observations for calculating a relative error of observations. If the number of samples of Data is smaller than minNr the error is calculated via the data set STD.
STD	data set for calculating the relative errors if in Data there are less rows per group than minNr. This replacement data set could for e.g. consist of reference standards with repeated measurement for each standard.

Value

data.frame or data.table with relative errors for each observation of 'Data'.

Author(s)

Solveig Pospiech

See Also

Other sub functions: check_readline, select.VarsElements

select.VarsElements   *select.VarsElements*

Description

The function returns a character vector of element abbreviations if the input object contained variables with element abbreviations. Input may be data.frame, matrix, character vector or named numeric. There are two options to use this functions:

- A) only one object

- B) with two objects

For A) the function checks for the pattern of element abbreviations, e.g. Al, S, Ca, etc. For B) the function checks for element abbreviations which are present in both objects. E.g. if x = c("Al", "Ba", "Ca") and y = c("Ba", "K", "Th") the return value will be "Ba". The resulting character vector is without duplicated entries, e.g. x = c("N", "P", "S", "S") results into c("N", "P", "S").

Usage

```
select.VarsElements(x, y, invert = FALSE)
```

Arguments

x	data.frame, character vector or named numeric containing element abbreviations as variables
y	optional, data.frame, character vector or named numeric containing element abbreviations as variables
invert	logical. If TRUE return variable names that do not match an element abbreviation pattern

Value

character vector of element abbreviations

Author(s)

Solveig Pospiech

See Also

Other sub functions: check_readline, relError_dataset

Examples

```
x = c("Al", "Ba", "Ca")
y = c("Ba", "K", "Th")
select.VarsElements(x, y)

myvector = c("Al", "Location", "Date", "S", "Ba", "OH")
select.VarsElements(myvector)
select.VarsElements(myvector, invert = TRUE)
```

Index