Package 'cnd'

March 31, 2020

Type Package

Title Compositional Nutrient Diagnosis

Version 1.0.1

Description The compositional nutrient diagnosis (CND) estimates imbalances in the nutrient composition of plant tissues, from the latter and its associated yield. The package aims to offer a platform on which to categorize and implement the different variants of the method. It also makes available the method developed by Landry in the context of the development of the Quebec's fertilization reference charts.

License GPL-3

Copyright Research and development institute for agri-environment (IRDA)

URL https://www.irda.qc.ca/en/

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports catenelson,
dplyr,
methods,
robustbase,
compositions,
testthat

Collate 'S4Class-CndCall.R'
'S4Class-CndData.R'
'S4Class-CndDataAugmented.R'
'S4Class-CndMethod.R'
'S4Class-CndNorm.R'
'cndAugment.R'
'S4Class-CndReference.R'
'S4Class-McdNorm.R'
'cnd.R'
'cndAnalysis.R'
'cndBind.R'
'cndCateNelson.R'
'cndCompositions.R'
'cndCutoff.R'
'cndMahalanobis.R'
'cndMcd.R'

'cndMethodLandry.R'
'cndReference.R'
'cndSubsetData.R'
'cndTransform.R'

R topics documented:

`cnd-package` *cnd: Compositional Nutrient Diagnosis*

Description

The compositional nutrient diagnosis (CND) estimates imbalances in the nutrient composition of plant tissues, from the latter and its associated yield. The method originates from Parent and Dhafir (1992) and has undergone a series of development since, which led to the appearance of multiple variants (Kihari et al., 2001; Parent et al., 2009; De Bauw et al., 2016; Parent et al., 2016). The `cnd` package aims to offer a platform on which to categorize and implement them. All variants, from Khiari et al. (2001) onward, share the same general steps to be performed sequentially:

`transformation` transform the composition data to fulfill requirements for the analysis (e.g. normality).

`subset` identify a subgroup of observations that represent balanced plants in term of nutrients, usually chosen among high yield observations.

`norm` find a norm (location and scatter) that caracterize this group. Although not originally interpreted in this way, the location (e.g. mean or median) could be thought as representing a perfect balance, while the scatter (e.g. variance or covariance) could be used to interpret interactions in the nutrient uptake by the plant.

`distance` define some distance, on the basis of the norm, that would define nutrient imbalances.

`analysis` compute the distance on the same, or another data set, to evaluate nutrient imbalances and perform supplementary analysis. As an example, Cates-Nelson analysis has often been used on the distance to caracterise the yield associated to balanced observations for the new dataset.

The package also makes available the method defined by Landry in the context of the development of the Quebec's fertilization reference charts (cndMethodLandry). The `examples` section cover all the important steps of the analysis, using the Landry's method.

Implementation

The package relies heavily on the S4 system which allow to define classes. A class possess specified fields (called slots) on which some properties can be defined and checked everytime an object of this class is created (note than slots are accessed using "@", and is the equivalent to "$" to access elements in `list` or columns in `data.frame`). S4 classes can be used as entries of S4 functions, thus ensuring some conditions are met before any computation occurs.

The package defines four basic S4 classes to be known to the user: `CndData` which contains all the data; `CndMethod` which contains specificities about the 5 components of the method; `CndNorm` that represent the norm to export; `CndReference` that contains a data subset and a norm. A `CndDataAugmented` class has also been defined to carry supplementary information with the data, while preserving properties of `CndData`.

The workflow to perform the cnd is as follow, see the `examples` section to put it into practice:

1. Define `CndData` and `CndMethod` objects.
2. Transform the data using the function `cndTransform` (use the method's component `transformation`). It takes `CndData` and `CndMethod` objects and return a `CndData` object.
3. Estimate a reference using the function `cndReference` (use the method's components `subset`, `norm` and `distance`). It takes the transformed `CndData` and the `CndMethod` objects and return a `CndReference` object, which contains a `CndNorm` object.
4. Estimate the distance and perform further analysis using `cndAnalysis` (use the method's components `distance` and `analysis`). It uses a new transformed `CndData` object, the `CndMethod` object and the estimated `CndNorm` object, and return a `CndDataAugmented` object.

Development of methods

A `CndMethod` object is composed of objects of class `CndCall`, one for each of the five method's component. Theses objects define the function and arguments to call by cnd functions. Each method's component can therefore be defined using your own function; the `CndMethod` and `CndCall` help pages explain how to define their input and output.

If new slots are required in the analysis, we suggest to create new classes that inherit the existing classes. The data flow, structure and validity checks would then be preserved as these objects also belong to the original classes. `CndDataAugmented` is an example of a general class built upon `CndData`, while `McdNorm` is a class built upon `CndNorm` that includes information specific to the MCD estimation. If new classes are defined, one might need to specify the behaviour of the functions `cndSubsetData` and `cndBind` for that class. In cases in which results need to be passed along data, they can be stored into the slots `suppl` or `other` of an augmented data object (`CndDataAugmented`).

The package use lower camel case for variable and function names (e.g. `transfData`, `cndAnalysis`) and upper cammel case for S4 class (e.g. `CndData`, `CndMethod`) as well as constructor functions for those classes (using the same names as the classes).

References

De Bauw P, Van Asten P, Jassogne L, Merckx R. 2016. Soil fertility gradients and production constraints for coffee and banana on volcanic mountain slopes in the East African Rift: A case study of Mt. Elgon. Agric. Ecosyst. Environ. 231: 166-175.

Khiari L, Parent L-E, Tremblay N. 2001. Selecting the high-yield subpopulation for diagnosing nutrient imbalance in crops. Agron. J. 93(4): 802-808.

Parent L-E, Dafir M. 1992. A theoretical concept of compositional nutrient diagnosis. J. Amer. Soc. Hort. Sci. 117(2): 239-242.

Parent L-E, Natale W, Ziadi N. 2009. Compositional nutrient diagnosis of corn using the Mahalanobis distance as nutrient imbalance index. Can. J. Soil Sci. 89(4): 383-390.

Parent S-E, Parent L, Rozane DE, Natale W. 2013. Plant ionome diagnosis using sound balances: case study with mango (Mangifera Indica). Front. Plant Sci. 4(article 449):1-12.

Author(s)

Maintainer: Alexandre Leblanc <`alexandre.leblanc@irda.qc.ca`>

Other contributors:

- Christine Landry <`christine.landry@irda.qc.ca`> [research team head]
- Anaïs Charles [research team member]
- Michèle Grenier [research team member]
- Gaétan Daigle [consultant]
- Research and development institute for agri-environment (IRDA) [copyright holder]

See Also

Useful links:

- https://www.irda.qc.ca/en/

Examples

```
#Generate random data for the example
n <- 50
yield <- 100 + rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1] #Alternative for black and white: label <- NULL

#Generate an object of class CndData.
cndData <- CndData(yield = yield, X = X, label = label)
```

```
#Generate an object of class CndMethod.
cndMethod <- cndMethodLandry(dropNutrient = "x3", labelName = "label")

#Transform the composition (X) of cndData the using cndMethod
transfData <- cndTransform(cndData, cndMethod)

#Estimate a reference from the transformed cndData using cndMethod
cndReference <- cndReference(transfData, cndMethod)
cndReference@norm

#Compute the distance and other analysis specified in cndMethod,
#using the estimated norm, on the same or a new dataset
cndAnalysis <- cndAnalysis(transfData, cndMethod, cndNorm = cndReference@norm)
cndAnalysis
```

`cndAnalysis` *Analyse a data set from a norm*

Description

Compute the distance on a new dataset, using a norm, and perform supplementary analysis.

Usage

```
cndAnalysis(transfData, cndMethod, cndNorm)
```

Arguments

`transfData`	an object of class `CndData`, with `X` already transformed.
`cndMethod`	an object of class `CndMethod`.
`cndNorm`	an object of class `CndNorm`.

Details

The slots `distance` of `cndMethod` make use of the `cndNorm` to compute distance of each observation of `transfData` from the centroid. The result is joined to `transfData`, usually as a column named `distance` under the slot `other`. The new object is then used to perform other analysis, specified by the slot `analysis` of `cndMethod`, which also append the result to `transfData`, usually in the `suppl` slot.

Value

Return a `CndDatAugmented` object corresponding to `transfData` augmented to also include the distance and the results of the supplementary analysis.

Examples

```
#Generate an object of class CndData.
n <- 50
yield <- 100+rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
cndData <- CndData(yield = yield, X = X)
```

```
#Compute a reference
cndMethod <- cndMethodLandry(dropNutrient = "x3")
transfData <- cndTransform(cndData, cndMethod)
cndReference <- cndReference(transfData, cndMethod)

#Perform the analysis
cndAnalysis(transfData, cndMethod, cndNorm = cndReference@norm)
```

`cndAugment` *Augment data*

Description

Join additional data to a `CndData` object under new slots. Information as data.frame, with the same number of row as slots of `CndData` can be added under the slot `other`, while supplementary data with a different structure must be added in the `suppl`.

Usage

```
cndAugment(cndData, other = data.frame(), suppl = list())
```

Arguments

`cndData`	an object of class `CndData` or `CndDataAugmented`.
`other`	a `data.frame` of additional values to associate with points of `cndData`, it must possess the same number of lines than `yield` and `X`.
`suppl`	`list` of supplementary information to pass along the object.

Details

If `cndData` is already an object of class `CndDataAugmented`, the entry `other` is binded as a column to the existing slot's `data.frame`, while the entry `suppl` is combined to the corresponding slot `list`.

Value

Return an object of class `CndDataAugmented` with the entries `other` and `suppl` added to the corresponding slots.

Examples

```
#General example
##Generate an object of class CndData
n <- 20
yield <- rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- data.frame(label1 = LETTERS[1:n], label2 = 1:n)
data <- CndData(yield = yield, X = X, label = label)

##Augment data of class CndData
dataAugmented <- cndAugment(data, other = X, suppl = list(comment = "comment"))
dataAugmented
```

```
##Augment data of class CndDataAugmented
cndAugment(dataAugmented, other = X, suppl = list(comment2 = "comment2"))
```

`cndBind` *Bind CndData*

Description

Bind slots of a `CndData`, or a `CndDataAugmented`, object into a single `data.frame`. For the latter, only the slot `other` is combined to the slots `X`, `Y` and `label`; the slot `suppl` is not used.

Usage

```
cndBind(cndData)
```

Arguments

`cndData`	An object of class `CndData` or `CndDataAugmented`.

Value

Return a single `data.frame`.

Examples

```
#' #General example
##Generate objects of classes CndData and CndDataAugmented
n = 20
yield <- rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
data <- CndData(yield = yield, X = X)
dataAugmented <- cndAugment(data, other = X, suppl = list(comment = "comment"))

##Bind elements into a data.frame for each class
cndBind(data)
cndBind(dataAugmented)
```

`cndCateNelson` *Cate Nelson analysis*

Description

Perform a Cate Nelson analysis on a `CndDataAugmented` object. The function is a wrapper of the function `cate_nelson` from the `catenelson` package. The `yield` slot is used as `y`, and the column `distance` of the slot `other` is used as `x`.

Usage

```
cndCateNelson(cndData, n_group = 2, trend = "negative",
  labelName = NULL, x_lab = "Squared distance", y_lab = "Yield", ...)
```

Arguments

`cndData`	An object of class `CndDataAugmented`, with a column named `distance` under the slot `other`, representing the squared distance from the centroid.
`n_group, trend, x_lab, y_lab`	`cate_nelson`'s parameter with imposed default values specific to cnd.
`labelName`	The column name of the `cndData`'s `label` slot to be use as the argument label in the function `cate_nelson`.
`...`	Other arguments to pass to the function `cate_nelson`.

Examples

```
#General data
##CndData
n <- 20
yield <- 100+rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]
cndData <- CndData(yield = yield, X = X, label = label)
transfData <- cndIlr(cndData)
##CndNorm and distance (CndDataAugmented)
cndNorm <- cndMcd(transfData, nSamp = 1000)
distance <- cndMahalanobis(transfData, cndNorm)

##Perform a Cate-Nelson analysis
cndCateNelson(distance, label = "label")
```

---

cndCompositions *Transformations from the package compositions*

Description

Wrapper of transformations `clr`, `ilr` and `alr` from the package compositions, to take `CndData` object as entry.

Usage

```
cndClr(cndData, dropNutrient = NULL, ...)

cndIlr(cndData, ...)

cndAlr(cndData, ...)
```

Arguments

`cndData`	an object of class `CndData`.
`dropNutrient`	`character`, the name of the nutrient (column name of the slot `X` of `CndData`) to drop after the transformation.
`...`	other parameters to pass to the original function of the package `compositions`.

Details

Only the requiered functions are called when needed, using the double colon symbol, to avoid namespace conflicts. The argument `dropNutrient` in `cndClr` has been added from the original function `clr`.

Value

All functions return a transformed `CndData` object.

Examples

```
#General data
##CndData
n <- 20
yield <- 100 + rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]
cndData <- CndData(yield = yield, X = X, label = label)

#Transform the compositional component of the data,
cndIlr(cndData)
cndClr(cndData, dropNutrient = "x3")
cndAlr(cndData)
```

`cndCutoff` *Yield cutoff subsetting*

Description

Subset cndData on the basis of a minimum yield value, either absolute or as a percentage.

Usage

```
cndCutoff(cndData, method, param)
```

Arguments

`cndData`	an object of class `CndData`.
`method`	`character`, defining the basis on which to subset data. Either `percent` or `value`.
`param`	`numeric`, specifying either the percentage of observertaion to keep (if `method = percent`), or the yield value delimiting the high yield subpopulation (if `method = value`).

Details

Only the number of observation encompassed by the percentage would be considered. For example, with 10 observations and a percentage to preserve of 25 percent, only the first two highest yield observations would be kept. For both method, equality is considered as part of the high yield subset (e.g. observation with a yield of 10000 would be kept in the subset if the cutoff value is also 10000).

Value

Return a subsetted `cndData` object.

Examples

```
#General data
##CndData
n <- 20
yield <- 100 + rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]
cndData <- CndData(yield = yield, X = X, label = label)

##Cutoff the data according to yield
cndCutoff(cndData, method = "percent", param = 50)
```

`cndMahalanobis` *Mahalanobis distance*

Description

Compute the Mahalanobis squared distance on the `X` slot of `cndData`, using `location` and `scatter` of `cndNorm`. The function is a wrapper of the function `mahalanobis` from the package `stats`.

Usage

```
cndMahalanobis(cndData, cndNorm, ...)
```

Arguments

`cndData`	an object of class `CndData`.
`cndNorm`	an object of class `CndNorm`.
`...`	other parameters to pass to the function `mahalanobis`.

Value

Return `cndData` augmented with the squared distance as a data.frame coulmn under the slot `other` (i.e. the returned object is of class CndDataAugmented).

Examples

```
#General data
##CndData
n <- 50L
yield <- 100+rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]
cndData <- CndData(yield = yield, X = X, label = label)
transfData <- cndIlr(cndData)

##CndNorm
cndNorm <- cndMcd(transfData, nSamp = 1000)

#Computing the distance
cndMahalanobis(transfData, cndNorm)
```

`cndMcd` *Minimum covariance determinant*

Description

Compute a robust norm (location and scatter) estimate via the minimum covariance determinant (MCD), on the `X` component of `cndData`. The function is a wrapper of the function `covMcd` from the package `robustbase`.

Usage

```
cndMcd(cndData, nSamp, ...)
```

Arguments

`cndData`	an object of class `CndData`.
`nSamp`	the number of subset used for initial estimates (`integer`) or the name of one of the method: `best`, `exact`, `deterministic`. See the function `covMcd` for more details on these methods.
`...`	other paramaters to pass to the function `covMcd` from the package `robustbase`.

Value

Return an object of class `McdNorm` that contains the same slots as `CndNorm`, but also the logarithm of the covariance matrix determinant (slot `logDet`) for the best subset (slot `subset`, returned as an object of class `CndData`).

Examples

```
#Generate an object of class CndData.
n <- 20
yield <- 100+rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
cndData <- CndData(yield = yield, X = X)
transfData <- cndClr(cndData, dropNutrient = "x3")
```

```
#Compute the MCD norm
cndMcd(transfData, nSamp = 1000)
```

cndMethodLandry *Landry's method*

Description

A predefined method developped by Landry in the context of the development of the Quebec's fertilization reference charts. Some flexibility is provided on the parameter values to consider during the analysis.

Usage

```
cndMethodLandry(dropNutrient = "fill", percent = 25, nSamp = 1000,
  n_group = 2, min_group_x = 2, min_group_y = 1, labelName = NULL)
```

Arguments

`dropNutrient`	(transformation) the nutrient to drop (column name of the slot `X`) after the clr transformation.
`percent`	(subset) the percentage of observation to include in the high yield subpopulation.
`nSamp`	(norm) the number of subsets used for initial estimates (`integer`) or the name of one of the method: `best`, `exact`, `deterministic`. See the function `covMcd` from the `robustbase` package for more details on these methods.
`n_group, min_group_x, min_group_y, labelName`	(analysis) arguments to pass to the function `cndCateNelson`: number of groups, minimum number of values per group in the x and y partioning, label's column name to use for the Cate-Nelson analysis.

Details

The steps of the predefined method, and some elements of context, are as follow:

`transformation` to normalize the data, `X` is first transformed as clr (function `cndClr`). However, because the this transformation makes the covariance matrix is not invertible, a condition required to compute the Mahalanobis distance, a nutrient is dropped after the transformation. The `fill` component is used by default, but any nutrient can be chosen by precising `dropNutrient` as an entry of the `cndMethodLandry` function. The approach was preconised by Parent et al. (2009).

`subset` a high yield subpopulation is then identified, using the function `cndCutoff`, at a percentage specified by the parameter `percent`. By default, the highest 25 percent in yield make this subset. Using a fixed percentage to define the high yield subpopulation was brought by De Bauw et al. (2016) to circumvent estimation problems in the approach developped by Khiari et al. (2001) and used in Parent el al. (2009).

`norm` a robust norm is then obtained through the minimum covariance determinant estimation (function `cndMcd`). The returned object is of class `McdNorm`, which inherits from the class `CndNorm`. The slot `subset` of the `McdNorm` object (representing further subseting by the MCD) can be used to associate a yield to the norm. However, this yield might not be representative for another dataset. The method, brought by Landry, replaces the iterative method of outlier detection used by Parent el al. (2009) to eliminate unbalanced cases that generate high yield (e.g. plants that grew in soils rich enough to reach maximum yield might be less influenced by balances).

`distance` the Mahalanobis distance (function `cndMahalanobis`) is then calculated, using the reference's norm, for the reference's data and on a new data set in the analysis. The distance was proposed by Parent et al. (2009), as an improvement from Khiari (2001), to eliminate colinearity.

`analysis` a Cate-Nelson analysis is then performed on the yield (y) and the squared distance (x), instead of a single element concentration. The method was used in both Khiari et al. (2001) and Parent et al. (2009) to classify points according to multiple elements at once, and might serve to delimit the yield and critical distance of observations near the centroid for data collected under new conditions.

References

De Bauw P, Van Asten P, Jassogne L, Merckx R. 2016. Soil fertility gradients and production constraints for coffee and banana on volcanic mountain slopes in the East African Rift: A case study of Mt. Elgon. Agric. Ecosyst. Environ. 231: 166-175.

Khiari L, Parent L-E, Tremblay N. 2001. Selecting the high-yield subpopulation for diagnosing nutrient imbalance in crops. Agron. J. 93(4): 802-808.

Parent L-E, Natale W, Ziadi N. 2009. Compositional nutrient diagnosis of corn using the Mahalanobis distance as nutrient imbalance index. Can. J. Soil Sci. 89(4): 383-390.

Examples

```
#Default method
method1 <- cndMethodLandry()

#The method with specific parameters
method2 <- cndMethodLandry(percent = 50, nSamp = 2000)
```

`cndReference` *Estimate the reference*

Description

Generate a reference object, applying the `subset`, `norm` and `distance` components of the method, that can be used for analysis on other data sets.

Usage

```
cndReference(transfData, cndMethod)
```

Arguments

`transfData`	an object of class `CndData`, with `X` already transformed.
`cndMethod`	an object of class `CndMethod`.

Details

First only preserve a `subset` of the data, then calculate the `norm` and compute the `distance` for this subset. Each step is specified by `cndMethod`.

Value

Return a `reference` object that contains the subset (a `CndDataAugmented` object that is also containing the distance) and the norm (of class `CndNorm`).

Examples

```
#General example
##Generate an object of class CndData
n <- 50
yield <- 100 + rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
cndData <- CndData(yield = yield, X = X)

##Identify the method and transform the data
cndMethod <- cndMethodLandry(dropNutrient = "x3")
transfData <- cndTransform(cndData, cndMethod)

##Compute the reference according to the defined method
cndReference(transfData, cndMethod)
```

`cndSubsetData` *Subset CndData*

Description

Subset the slots `yield`, `X` and `label` of a `CndData` object, as well as the slot `other` for a `CndDataAugmented` object.

Usage

```
cndSubsetData(cndData, subset)
```

Arguments

`cndData`	an object of class `CndData` or `CndDataAugmented`.
`subset`	`logical` indicating which row to keep.

Examples

```
#General example
##Generate an object of class CndData and CndDataAugmented
n <- 20
yield <- rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
data <- CndData(yield = yield, X = X)
dataAugmented <- cndAugment(data, other = X, suppl = list(comment = "comment"))
```

```
##Subset both data sets
subset <- sample(c(TRUE,FALSE), n, replace = TRUE)
cndSubsetData(data, subset)
cndSubsetData(dataAugmented, subset)
```

cndTransform *Transform CndData*

Description

Perform the `transformation`, specified by `cndMethod`, on the `X` component of the `CndData` object.

Usage

```
cndTransform(cndData, cndMethod)
```

Arguments

`cndData`	an object of class `CndData`.
`cndMethod`	an object of class `CndMethod`.

Value

Return a `CndData` object of the transformed data.

Examples

```
#Generate random data for the example
n <- 50
yield <- 100 + rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1] #Alternative for black and white: label <- NULL

#Generate an object of class CndData.
cndData <- CndData(yield = yield, X = X, label = label)

#Generate an object of class CndMethod.
cndMethod <- cndMethodLandry(dropNutrient = "x3", labelName = "label")

#Transform the composition (X) of cndData the using cndMethod
cndTransform(cndData, cndMethod)
```

S4Class-CndCall *CND Call (S4 Class)*

Description

S4 class that contains the function name and arguments to be use in a method, see `CndMethod`.

Usage

```
CndCall(fun = character(0), args = list())
```

Details

The function `CndCall` should be used to build an object.

Slots

`fun` the name of the function to call.

`args` a `list` of argument to pass to the function. Cnd objects used as parameter in `fun`, such as `cndData` and `cndNorm` should not be part of `args`.

Examples

```
#Examine slots for the class
getSlots("CndCall")

#General example (illustrating the call for a transformation)
transformation <- CndCall(fun = "cndClr", args = list(dropNutrient = "x3"))
```

S4Class-CndData *CND Data (S4 class)*

Description

S4 class that contains information on the data to use for the cnd analysis.

Usage

```
CndData(yield = data.frame(yield = numeric(0)), X = data.frame(),
  label = data.frame())
```

Details

The validity method check if: `yield` only possesses one column; `yield`, `X` and `label` have the same number of lines; columns of `yield` and `X` are of class `numeric`.

The function `CndData` should be used to build an object. Both arguments `yield` and `label` can be provided as vectors, they will be coerced into `data.frame`. If provided as vector, the column name of `label` would be "label", and the one `yield` would be "yield". For the latter, the column name would also be overwriten as "yield" if the entry is a `data.frame`. If empty, the number of rows of `label` would be matched to those of `yield` and `X` at initialisation.

Slots

`yield` a `data.frame`, with one column, of yield associated to lines of `X`. The entry can be a `numeric` vector if the object is built using the function `CndData`.

`X` a `data.frame` of composition, with columns corresponding to nutrients and lines to samples.

`label` (optional) `data.frame` of labels associated to lines of `X`. By default, an empty `data.frame` with the same number of rows as `yield` and `X`, but no columns.

Examples

```
#Examine slots for the class
getSlots("CndData")

#General example
##Generate an object of class CndData
n <- 20
yield <- rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]

##Generate an object of class CndData.
data <- CndData(yield = yield, X = X, label = label)

#Examples with zero or multiple label columns
##Zero columns
data <- CndData(yield = yield, X = X)

##Two columns
label <- data.frame(label1 = label, label2 = 1:n)
data <- CndData(yield = yield, X = X, label = label)
```

`S4Class-CndDataAugmented` *CND Data Augmented (S4 class)*

Description

S4 class that contains information on the data to use for the cnd analysis, but that also contains additional fields.

Usage

```
CndDataAugmented(yield = data.frame(yield = numeric(0)),
  X = data.frame(), label = data.frame(), other = data.frame(),
  suppl = list())
```

Details

The validity method check if: `yield` only possesses one column; `yield`, `X`, `label` and `other` have the same number of lines; columns of `yield` and `X` are of class `numeric`.

The function `CndDataAugmented` should be used to build an object. Both arguments `yield` and `label` can be provided as vectors, they will be coerced into `data.frame`. If provided as vector, the column name of `label` would be "label", and the one `yield` would be "yield". For the latter, the column name would also be overwriten as "yield" if the entry is a `data.frame`. If empty, the number of rows of `label` would be matched to those of `yield` and `X` at initialisation.

Slots

`yield` `data.frame`, with one column, of yield associated to lines of `X`. The entry can be a `numeric` vector if the object is built using the function `CndData`.

`X` `data.frame` of composition, with columns corresponding to nutrients and lines to samples.

`label` (optional) `data.frame` of labels associated to lines of `X`. By default, an empty `data.frame` with the same number of rows as `yield` and `X`, but no columns.

`other` a `data.frame` of additional values to associate with points of `cndData`, it must possess the same number of lines than `yield` and `X`.

`suppl` `list` of supplementary material to pass along the object.

Examples

```
#General example
#' ##Observe slots of CndMethod
getSlots("CndDataAugmented")

##Generate data for the example
n <- 20
yield <- rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]

##Generate an object of class CndDataAugmented
data <- CndDataAugmented(yield = yield, X = X, label = label,
other = X, suppl = list(comment = "some comment"))
```

S4Class-CndMethod *CND Method (S4 class)*

Description

S4 class that contains information of the method to use throughout the `cnd`. The function `CndMethod` should be used to build an object.

Usage

```
CndMethod(transformation, subset, norm, distance, analysis)
```

Details

A method is made of cndCall, one for each part of the analysis. The definition of functions associated to the `CndCall` of `transformation`, `subset`, `norm` and `analysis` must have `cndData` as a parameter. In addition to `cndData`, the definition of function of `distance` must also have `cndNorm` as a parameter. However, these cnd object must not be indicated in the `args` of each of the `CndCall` as they are already assigned in functions `cndTransform`, `cndReference` and `cndAnalysis`.

The function defined in `transformation`, `subset`, `distance` and `analysis` should all return a `CndData` object, or an object that inherit from `CndData`. We recommand that the function associated to `distance` return a `cndDataAugmented` object with a column named `distance` under the slot `other`; although not mandatory, it is the expected entry for a Cate-Nelson analysis (`cndCateNelson`). The function in `norm`, should return a `CndNorm` object or an object of a class that inherit from this class.

The function `cndMethodLandry` call a predefined method as a whole. New methods can also be built by assembling predefined cnd functions for each section. Those already defined are as follow.

`transformation` the functions `cndClr`, `cndAlr` and `cndIlr`.

`subset` only the `cndCutoff` method is presently defined. When developping new methods, the function `cndSubsetData` can be used once we know the observation to keep, to subset the `CndData` object.

`norm` only the `cndMcd` method is presently defined.

`distance` only the `cndMahalanobis` distance is presently defined.

`analysis` only the `cndCateNelson` analysis is presently defined.

Slots

`transformation` a `CndCall` object defining the function and arguments to use in `cndTransform`.

`subset` a `CndCall` object defining the function and arguments to use for a first subsetting of data in `cndReference`.

`norm` a `CndCall` object defining the function and arguments to use to define the norm in `cndReference`

`distance` a `CndCall` object defining the function and arguments that define the distance to compute in `cndReference` and `cndAnalysis`

`analysis` a `CndCall` object defining the function and arguments to use for supplementary analysis in `cndAnalysis`

Examples

```
#General example
##Observe slots of CndMethod
getSlots("CndMethod")

##Define elements of the method
transformation <- CndCall("cndClr", list(dropNutrient = "x3"))
subset          <- CndCall("cndCutoff", args = list(method = "percent", param = 50))
norm            <- CndCall("cndMcd", args = list(n = 100))
distance        <- CndCall("cndMahalanobis", args = list())
analysis       <- CndCall("cndCateNelson", args = list(min_group_x = 5, min_group_y = 5))
```

```
##Generate an object of class CndMethod.
method         <- CndMethod(transformation = transformation,
subset = subset, norm = norm, distance = distance, analysis = analysis)
```

S4Class-CndNorm *CND Norm (S4 Class)*

Description

S4 class that contains location and scatter estimates.

Usage

```
CndNorm(location = numeric(0), scatter = matrix(numeric(0), 0, 0))
```

Details

The validity method check if: `location` is of class `numeric`; `scatter` is a numerical matrix; the location length correspond to the number of row and column of scatter.

The function `CndNorm` can be used to build an object.

Slots

`location` a `numeric`, location estimate.

`scatter` a `matrix`, variance or covariance matrix estimate.

Examples

```
#Examine slots for the class
getSlots("CndNorm")

#Define the location and scatter
location <- 1:3
scatter <- matrix(1:9,3,3)

#Generate an object of class CndNorm
norm <- CndNorm(location = location, scatter = scatter)
```

S4Class-CndReference *CND Reference (S4 Class)*

Description

S4 class that contains the subset selected prior to the norm calculation as well as the norm. It might be used to perform analysis on another dataset. The function `CndReference` should be used to build an object.

Slots

`subset` a `CndData` object representing a selection of data prior to the norm calculation. Does not includes further selection while calculating the norm.

`norm` a `CndCNorm` object associated to the subset.

Examples

```
#Examine slots for the class
getSlots("CndReference")

#General example
##Generate an object of class CndData
n <- 50
yield <- 100 + rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]
cndData <- CndData(yield = yield, X = X, label = label)

##Generate an object of class CndNorm
##Define the location and scatter
location <- 1:3
scatter <- matrix(1:9,3,3)
cndNorm <- CndNorm(location = location, scatter = scatter)

##Generate an object of class CndReference
reference <- CndReference(subset = cndData, norm = cndNorm)
```

S4Class-McdNorm *CND Minimum covariance determinant estimates (S4 Class)*

Description

S4 class that extend the class `CndNorm` to also include the logarithm of the covariance matrix determinant `logDet` associated to the robust method.

Usage

```
McdNorm(location = numeric(0), scatter = matrix(numeric(0), ncol = 0,
  nrow = 0), logDet = numeric(0), subset = CndData())
```

Details

The validity method check if: `location` and `scatter` respect validity checks defined for `CndNorm` and if `logDet` only possess one value.

The function `McdNorm` can be used to build an object.

Slots

`location` a `numeric`, location estimate.

`scatter` a `matrix`, variance or covariance matrix estimate.

`logDet` a `matrix`, logarithm of the determinant of the covariance matrix, which was minimized in the mcd.

`subset` a `CndData` object, representing the subset of observation that minimized the determinant, and used to establish the location and scatter.

Examples

```
#Examine slots for the class
getSlots("McdNorm")

#General example
##Define elements provided in by the mcd
###The norm and logDet
location <- 1:3
scatter <- matrix(1:9,3,3)
logDet <- 1

##A data subset (here some CndData object)
n <- 10
yield <- rnorm(n)
X <- data.frame(x1 = runif(n), x2 = runif(n), x3 = runif(n))
label <- LETTERS[(seq_len(n)-1)%%4+1]
subset <- CndData(yield = yield, X = X, label = label)

##Generate an object of class McdNorm
norm <- McdNorm(location = location, scatter = scatter,logDet = logDet, subset = subset)
```

Index