Package 'ConfMatrix'

May 30, 2026

Type Package

Title Confusion Matrix

Version 0.2.1

Author Francisco Javier Ariza-Lopez [aut],
Paola Barba-Ceballos [aut],
Silverio Vilchez-Lopez [aut, cre],
Jose Rodriguez-Avi [aut],
Maria Virtudes Alba-Fernandez [aut],
Jose Luis Garcia-Balboa [aut]

Maintainer Silverio Vilchez-Lopez <svilchez@ujaen.es>

Description Thematic quality indices are provided to facilitate the evaluation and quality control of geospatial data products (e.g. thematic maps, remote sensing classifications, etc.). The indices offered are based on the so-called confusion matrix. This matrix is constructed by comparing the assigned classes or attributes of a set of pairs of positions or objects in the product and the ground truth. In this package it is considered that the classes of the ground truth correspond to the columns and that the classes of the product to be valued correspond to the rows. The package offers two object classes with their methods: 'ConfMatrix' (Confusion matrix) and 'QCCS' (Quality Control Columns Set). The 'ConfMatrix' class of objects offers more than 20 methods based on the confusion matrix. The 'QCCS' class of objects offers a different perspective in which the ground truth is considered to allow the values of the column marginals to be fixed, see Ariza López et al. (2019) <doi:10.3390/app9204240> and Canran Liu et al. (2007) <doi:10.1016/j.rse.2006.10.010> for more details. The package was created with 'R6'.

Imports R6, Rdpack, ggplot2, gridExtra

RdMacros Rdpack

License GPL

Encoding UTF-8

Suggests knitr, rmarkdown, roxygen2, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0
NeedsCompilation no
Repository CRAN
Date/Publication 2026-05-29 22:50:02 UTC

Contents

ConfMatrix-package 2
ConfMatrix 3
QCCS 51

Index **57**

`ConfMatrix-package` *ConfMatrix: Statistical Tools for Thematic-Accuracy Quality Control*

Description

Statistical tools for validating thematic cartography using confusion matrices and column-wise quality control.

Details

The package is organized into two main classes. Use the following links to access the reference documentation for each method:

`ConfMatrix` **Class (Global and Class-level Evaluation):**

- **Global Indices:** `Overall Accuracy (OA)`, `Kappa`, `Tau`, `Ji.test`, `Relative_Agreement_Index`.
- **Class-specific (i):** `UserAcc_i`, `ProdAcc_i`, `Error_i`, `f_i`, `g_i`, `h_i`.
- **Averages and Combined:** `AvUserAcc`, `AvProdAcc`, `AvError`, `CombUserAcc_i`, `CombProdAcc_i`.
- **Advanced Tests:** `Marghit` (Marginal Homogeneity), `Quasi_Independence`.

`QCCS` **Class (Quality Control):**

- **Tests:** `Exact.test`, `Ji.test`, `JiGlobal.test`.

Author(s)

Maintainer: Silverio Vilchez-Lopez <svilchez@ujaen.es>

Authors:

- Silverio Vilchez-Lopez <svilchez@ujaen.es>
- Francisco Javier Ariza-Lopez <fjariza@ujaen.es>
- Paola Barba-Ceballos <pbarba@ujaen.es>
- Jose Rodriguez-Avi <jravi@ujaen.es>
- Maria Virtudes Alba-Fernandez <mvalba@ujaen.es>
- Jose Luis Garcia-Balboa <jlbalboa@ujaen.es>

`ConfMatrix` *Confusion matrix Class*

Description

The `ConfMatrix` class works with confusion matrices, thus providing the possibility of calculating several indices with their corresponding variances and confidence intervals. A confusion matrix is constructed by comparing a sample of a set of common positions in the product and the ground truth. Appropriate sampling methods must be applied to generate the confusion matrix. It is considered that the classes of the ground truth correspond to the columns and that the classes of the product to be valued correspond to the rows. First, an object of this class of object must be created (instantiated) and then the methods that offer the index calculations will be invoked. Mnemonic method names are proposed and are therefore long, for example methods that provide averages start with "AV" and those that provide combinations start with "Comb". Methods related to a specific thematics class end with the ending "_i".

Error Messages

List of possible errors:

- `Error type 1`: Non-square matrix.
- `Error type 2`: Single element matrix.
- `Error type 3`: Negative values.
- `Error type 4`: Sum of elements 0.
- `Error type 5`: Sum of rows 0.
- `Error type 6`: Sum of columns 0.
- `Error type 7`: It is not a matrix.

Mathematical elements

- x_{ii}: diagonal element of the matrix.
- x_{ij}: element i, j of the matrix.
- x_{i+}: sum of all elements in rows i.
- x_{+j}: sum of all elements in column j.
- M: number of classes.
- $\overline{x}_{i+}$: sum of all elements of row i except element i of the diagonal.
- $\overline{x}_{+i}$: sum of all elements of column i except element i of the diagonal.
- N_{Total}:Total count of elements in the instance's Confusion Matrix.

$$N_{Total} = \sum_{i,j}^{M} x_{ij}$$

- N_i/N_j: Total count of elements in row i or column j.

$$N_i = x_{i+}$$

$$N_j = x_{+j}$$

- N_{ij}: Total count of elements in row i and column j.

$$N_{ij} = x_{i+} + x_{+j} - x_{ii}$$

Public fields

`Values matrix`. Matrix of integer values.

`ID character`. Identifier (maximum 50 characters).

`Date Date`. Reference date.

`ClassNames character`. Name of the classes (maximum 20 characters).

`Source` Source `character` | NULL. Indicates where the matrix comes from.

Methods

Public methods:

- `ConfMatrix$new()`
- `ConfMatrix$plot.index()`
- `ConfMatrix$plot.UserProdAcc()`
- `ConfMatrix$print()`
- `ConfMatrix$AllParameters()`
- `ConfMatrix$UserAcc()`
- `ConfMatrix$UserAcc_i()`
- `ConfMatrix$AvUserAcc()`
- `ConfMatrix$CombUserAcc()`
- `ConfMatrix$ProdAcc()`
- `ConfMatrix$ProdAcc_i()`
- `ConfMatrix$AvProdAcc()`
- `ConfMatrix$CombProdAcc()`
- `ConfMatrix$UserProdAcc()`
- `ConfMatrix$CombUserProdAcc()`
- `ConfMatrix$AvUserProdAcc()`
- `ConfMatrix$AvUserProdAcc_i()`
- `ConfMatrix$UserProdAcc_W()`
- `ConfMatrix$OverallAcc()`
- `ConfMatrix$Kappa()`
- `ConfMatrix$ModKappa()`
- `ConfMatrix$UserKappa_i()`
- `ConfMatrix$ModKappaUser_i()`
- `ConfMatrix$ProdKappa_i()`

- ConfMatrix$ModKappaProd_i()
- ConfMatrix$DetailKappa()
- ConfMatrix$DetailCondKappa()
- ConfMatrix$DetailWKappa()
- ConfMatrix$Tau()
- ConfMatrix$DetailWTau()
- ConfMatrix$Ent()
- ConfMatrix$AvNormEnt()
- ConfMatrix$GeomAvNormEnt()
- ConfMatrix$AvMaxNormEnt()
- ConfMatrix$EntUser_i()
- ConfMatrix$NormEntUser()
- ConfMatrix$EntProd_i()
- ConfMatrix$NormEntProd()
- ConfMatrix$Sucess()
- ConfMatrix$Sucess_i()
- ConfMatrix$AvHellAcc()
- ConfMatrix$AvHellAcc_i()
- ConfMatrix$AvShortAcc()
- ConfMatrix$ShortAcc_i()
- ConfMatrix$GroundTruth()
- ConfMatrix$GroundTruth_i()
- ConfMatrix$HellingerDist()
- ConfMatrix$QES()
- ConfMatrix$MTypify()
- ConfMatrix$MBootStrap()
- ConfMatrix$MNormalize()
- ConfMatrix$MPseudoZeroes()
- ConfMatrix$OverallAcc.test()
- ConfMatrix$Kappa.test()
- ConfMatrix$Tau.test()
- ConfMatrix$TSCM.test()
- ConfMatrix$QIndep.test()

`ConfMatrix$new()`**:** Public method to create an instance of the `ConfMatrix` class. When creating it, values must be given to the matrix. The values of the matrix must be organized in such a way that the columns represent the classes in the reference and the rows represent the classes in the product being evaluated. The creation of a `ConfMatrix` instance includes a series of checks on the data. If checks are not met, the system generates coded error messages. The optional possibility of adding metadata to the matrix is offered.

Usage:

```
ConfMatrix$new(
  Values,
  ID = NULL,
  Date = NULL,
  ClassNames = NULL,
  Source = NULL
)
```

Arguments:

`Values` Matrix of integer values. A matrix must be added.

`ID` Unique identifier (Optional). It is a character string with a maximum length of 50 characters. By default, CM_i will be taken as identification. Where i will be the number of `ConfMatrix` instances already defined in the session.

`Date` Date provided by the user in format "YYYY-MM-DD" (Optional). By defaults `Sys.Date()`.

`ClassNames` Name of the classes (Optional). It is given by a character strings vector whose elements are the name of the classes. Each element of the vector is a string of maximum 20 characters. By default for the column elements they will be PC_i (Producer class) and for the elements of row UC_i (User class), with i being the corresponding row or column number.

`Source` (Optional) Indicates where the matrix comes from (article, project, etc.). It is suggested to enter a reference or a DOI. A character string with a maximum length of 80 characters can be entered. By default, is NULL.

Returns: Object of the `ConfMatrix` class, or an error message.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
cm<-ConfMatrix$new (A,ID="5",Date="27-10-2023",Source="Congalton and Green,
2008")
```

`ConfMatrix$plot.index()`**:** Public method that provides a graph of the indices of the functions ConfMatrix$OverallAcc, ConfMatrix$Kappa, ConfMatrix$Tau, ConfMatrix$AvHellAcc and ConfMatrix$AvShortAcc with their corresponding standard deviations.

Usage:

```
ConfMatrix$plot.index()
```

Returns: A graph of the indices of the functions OverallAcc, Kappa, Tau, AvHellAcc, AvShortAcc with their corresponding standard deviations.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$plot.index()
```

`ConfMatrix$plot.UserProdAcc()`**:** Public method that provides a graph for the user's and producer's accuracies and standard deviations.

Usage:

```
ConfMatrix$plot.UserProdAcc()
```

Returns: The graph of the accuracy index of users and producers with their corresponding standard desviation.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$plot.UserProdAcc()
```

`ConfMatrix$print()`**:** Public method that shows all the data entered by the user for a instance.

Usage:

```
ConfMatrix$print()
```

Returns: ConfMatrix object identifier, date, class name, data source and confusion matrix.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,ClassNames=c("Deciduous","conifer","agriculture",
"shrub"),Source="Congalton and Green 2008")
p$print()
```

`ConfMatrix$AllParameters()`**:** Public method in which multiple parameters are calculated for the given confusion matrix. This method is equivalent to ConfMatrix$OverallAcc,ConfMatrix$UserAcc, ConfMatrix$ProdAcc,ConfMatrix$Kappa and ConfMatrix$MPseudoZeroes.

Usage:

```
ConfMatrix$AllParameters()
```

Returns: The following list of elements: the confusion matrix, dimension, total sum of cell values, overall accuracy, overall accuracy variance, global kappa index, global kappa simplified variance, producer accuracy by class, user accuracy by class, and pseudoceros matrix.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AllParameters()
```

`ConfMatrix$UserAcc()`**:** Public method for deriving the index called user's accuracy for all the classes in a `ConfMatrix` object instance. The user's accuracy for the class i of a thematic product is calculated by dividing the value in the diagonal of class i by the sum of all values in the row of the class i (row marginal).

The method also offers the variance and confidence interval. The reference Congalton and Green (2008) is followed for the computations.

$$UserAcc = \frac{x_{ii}}{x_{i+}}$$

$$\sigma^2_{UserAcc} = \frac{UserAcc \cdot (1 - UserAcc)}{N_i}$$

Usage:

```
ConfMatrix$UserAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Returns: A list of vectors, containing the user's accuracy real values for all classes, their variances and confidence intervals for each class, respectively.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$UserAcc()
```

`ConfMatrix$UserAcc_i()`**:** Public method for deriving the index called user's accuracy for a specific class i in a ConfMatrix object instance. The user's accuracy for the class i of a thematic product is calculated by dividing the value in the diagonal of class i by the sum of all values in the row of the class i (row marginal). The method also offers the variance and confidence interval. The reference Congalton and Green (2008) is followed for the computations.

Usage:

```
ConfMatrix$UserAcc_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Details:

$$UserAcc_i = \frac{x_{ii}}{x_{i+}}$$

$$\sigma^2_{UserAcc_i} = \frac{UserAcc_i \cdot (1 - UserAcc_i)}{N_i}$$

Returns: A list of real values containing the user's accuracy for class i, its variance, and its confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$UserAcc_i(2)
```

`ConfMatrix$AvUserAcc()`**:** Public method that provides the arithmetic average, without weighing, of all user's accuracies of a ConfMatrix object instance. The method also offers the variance and confidence interval. The reference Tung and LeDrew (1988) is followed for the calculations.

Usage:

```
ConfMatrix$AvUserAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$AvUserAcc = \frac{1}{M} \sum_{i=1}^{M} \frac{x_{ii}}{x_{i+}}$$

$$\sigma^2_{AvUserAcc} = \frac{AvUserAcc \cdot (1 - AvUserAcc)}{N_{Total}}$$

Returns: A list of real values containing the average user's accuracy, its variance, and its confidence interval.

Examples:

```
A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$AvUserAcc()
```

`ConfMatrix$CombUserAcc()`**:** Public method that provides the combined user's accuracy. Which is the average of the overall accuracy and the average user's accuracy. The method also offers the variance and confidence interval. The reference Tung and LeDrew (1988) is followed for the calculations.

Usage:

```
ConfMatrix$CombUserAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$CombUserAcc = \frac{OverallAcc + AvUserAcc}{2}$$

$$\sigma^2_{CombUserAcc} = \frac{CombUserAcc \cdot (1 - CombUserAcc)}{N_{Total}}$$

where:

- $OverallAcc$: overall accuracy.

Returns: A list of real values containing the combined accuracy from the user's perspective, its variation and confidence interval.

Examples:

```
A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$CombUserAcc()
```

`ConfMatrix$ProdAcc()`**:** Public method for deriving the index called producer's accuracy for all the classes in a ConfMatrix object instance. The producer's accuracy for the class i of a thematic product is calculated by dividing the value in the diagonal of class i by the sum of all values in the row of the class i (column marginal). The method also offers the variance and confidence interval. The reference Congalton and Green (2008) if followed for the computations.

Usage:

```
ConfMatrix$ProdAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$ProdAcc = \frac{x_{ii}}{x_{+j}}$$

$$\sigma^2_{ProdAcc} = \frac{ProdAcc \cdot (1 - ProdAcc)}{N_j}$$

Returns: A list of vectors each one containing the producer's accuracy real values for all classes, their variances and confidence intervals for each class, respectively.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$ProdAcc()
```

`ConfMatrix$ProdAcc_i()`**:** Public method for deriving the index called producer's accuracy for a specific class i in a ConfMatrix object instance. The user's accuracy for the class i of a thematic product is calculated by dividing the value in the diagonal of class i by the sum of all values in the column of the class i (column marginal). The method also offers the variance and confidence interval. The reference Congalton and Green (2008) is followed for the calculations.

Usage:

```
ConfMatrix$ProdAcc_i(i, alpha = NULL)
```

Arguments:

`i` Producer class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Details:

$$ProdAcc_i = \frac{x_{ii}}{x_{+j}}$$

$$\sigma^2_{ProdAcc_i} = \frac{ProdAcc_i \cdot (1 - ProdAcc_i)}{N_j}$$

Returns: A list of real values containing the producer's accuracy for class i, its variance, and its confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$ProdAcc_i(1)
```

`ConfMatrix$AvProdAcc()`**:** Public method that provides the arithmetic average of all producer's accuracies of a ConfMatrix object instance. The method also offers the variance and confidence interval. The reference Tung and LeDrew (1988) is followed for the calculations.

Usage:

```
ConfMatrix$AvProdAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$AvProdAcc = \frac{1}{M} \sum_{i=1}^{M} \frac{x_{ii}}{x_{+j}}$$

$$\sigma^2_{AvProdAcc} = \frac{AvProdAcc \cdot (1 - AvProdAcc)}{N_{Total}}$$

Returns: A list of real values containing the average producer's accuracy, its variance, and its confidence interval.

Examples:

```
A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$AvProdAcc()
```

`ConfMatrix$CombProdAcc()`: Public method that provides the combined producer's accuracy. Which is the average of the overall accuracy and the average producer accuracy. The method also offers the variance and confidence interval. The reference Tung and LeDrew (1988) is followed for the calculations.

Usage:

```
ConfMatrix$CombProdAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$CombProdAcc = \frac{OverallAcc + AvProdAcc}{2}$$

$$\sigma^2_{CombProdAcc} = \frac{CombProdAcc \cdot (1 - CombProdAcc)}{N_{Total}}$$

where:

- $OverallAcc$: overall accuracy.
- $AvProdAcc$: average accuracy from producer's perspective.

Returns: A list of real values containing the combined accuracy from producer's perspective, its variance and confidence interval.

Examples:

```
A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$CombProdAcc()
```

`ConfMatrix$UserProdAcc()`: Public method that calculates the user's and the producer's indexes jointly. This method is equivalent to the methods ConfMatrix$UserAcc and ConfMatrix$ProdAcc.

Usage:

```
ConfMatrix$UserProdAcc()
```

Returns: A list containing the producer's and user's accuracies and their standard deviations, respectively.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$UserProdAcc()
```

`ConfMatrix$CombUserProdAcc()`: Public method that provides the combined accuracy, defined by the average of the overall accuracy and the Hellden's average accuracy, which refers to the average user's and producer's accuracies. The method also offers the variance and confidence interval. The reference Liu et al. (2007) is followed for the calculations.

Usage:

```
ConfMatrix$CombUserProdAcc(a = NULL)
```

Arguments:

a
: Significance level. By default 0.05.

Details:

$$CombUserProdAcc = \frac{OverallAcc + AvHellAcc}{2}$$

$$\sigma^2_{CombUserProdAcc} = \frac{CombUserProdAcc \cdot (1 - CombUserProdAcc)}{N_{Total}}$$

where:

- $OverallAcc$: overall accuracy.
- $AvHellAcc$: average of Hellden's mean accuracy index.

Returns: A list of real values containing the combined accuracy from both user's and producer's perspectives, its variance and confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$CombUserProdAcc()
```

`ConfMatrix$AvUserProdAcc()`**:** Public method that provides the arithmetic average of all user's and producer's accuracy indexes of a ConfMatrix object instance. The method also offers the variance and confidence interval. The reference Liu et al. (2007) is followed for the calculations.

Usage:

```
ConfMatrix$AvUserProdAcc(alpha = NULL)
```

Arguments:

alpha Significance level. By default 0.05.

Details:

$$AvUserProdAcc = \frac{AvUserAcc + AvProdAcc}{2}$$

$$\sigma^2_{AvUserProdAcc} = \frac{AvUserProdAcc \cdot (1 - AvUserProdAcc)}{N_{Total}}$$

where:

- $AvUserAcc$: average accuracy from user's perspective.
- $AvProdAcc$: average accuracy from producer's perspective.

Returns: A list of real values containing the average mean precision values from the user's and producer's perspective, their variance and confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc()
```

`ConfMatrix$AvUserProdAcc_i()`: Public method that provides the average of user's and producer's accuracies for a specific class i. The method also offers the variance and confidence interval. The reference Liu et al. (2007) is followed for the calculations.

$$AvUserProdAcc_i = \frac{UserAcc_i + ProdAcc_i}{2}$$

$$\sigma^2_{AvUserProdAcc_i} = \frac{AvUserProdAcc_i \cdot (1 - AvUserProdAcc_i)}{N_{ij}}$$

where:

- $UserAcc_i$: user accuracy index for class i.
- $ProdAcc_i$: producer accuracy index for class i.

Usage:

```
ConfMatrix$AvUserProdAcc_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing the average of user's and producer's accuracies, its variance and confidence interval for class i.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc_i(2)
```

`ConfMatrix$UserProdAcc_W()`: Public method that calculates the weighted user's, producer's and overall accuracies and their standard deviations. The reference Congalton and Green (2008) is followed for the computations.
Be

$$Overall_W = \frac{\sum_{i=1}^{M} p_{ii}}{\sum_{i,j=1}^{M} p_{ij}}$$

where $p_{ij} = \frac{x_{ij}}{\sum_{i,j=1}^{M} x_{ij}}$

$$UserAcc_W = \frac{p_{o_{i+}}}{p_{i+}}$$

$$ProdAcc_W = \frac{p_{o_{+j}}}{p_{+j}}$$

$$\sigma^2_{UserAcc_W} = \frac{UserAcc_W \cdot (1 - UserAcc_W)}{N_i}$$

$$\sigma^2_{ProdAcc_W} = \frac{ProdAcc_W \cdot (1 - ProdAcc_W)}{N_j}$$

where $p_o = \sum_{i,j=1}^{M} w_{ij} p_{ij}$ and $0 \leq w_{ij} \leq 1$ for $i \neq j$ and $w_{ii} = 1$ for $i = j$.

Usage:

`ConfMatrix$UserProdAcc_W(WM)`

Arguments:

`WM` Weight matrix (as matrix)

Returns: A list with the weight matrix, the product of the confusion matrix and the weight matrix, overall, user and producer weighted accuracies and their standard deviations.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
WM<- t(matrix(c(1,0,0.67,1,0,1,0,0,1,0,1,1,0.91,0,0.61,1),nrow=4,ncol=4))
p$UserProdAcc_W(WM)
```

`ConfMatrix$OverallAcc()`**:** Public method to calculate the global index called Overall Accuracy. The Overall Accuracy is calculated by dividing the sum of the entries that form the major diagonal (i.e., the number of correct classifications) by the total number of cases. The method also offers the variance and confidence interval. The reference Congalton and Green (2008) is followed for the computations.

$$OverallAcc = \frac{\sum_{i=1}^{M} x_{ii}}{\sum_{i,j=1}^{M} x_{ij}}$$

$$\sigma^2_{OverallAcc} = \frac{OverallAcc \cdot (1 - OverallAcc)}{N_{Total}}$$

Usage:

`ConfMatrix$OverallAcc(alpha = NULL)`

Arguments:

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing the overall accuracy, its variance, and its confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A)
p$OverallAcc()
```

`ConfMatrix$Kappa()`**:** Public method that provides Kappa coefficient, which measures the relationship between the observed proportion of agreement and the proportion expected to occur by chance. The method also offers the variance and confidence interval. The reference Cohen (1960) is followed for the calculations.

Usage:

`ConfMatrix$Kappa(alpha = NULL)`

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$Kappa = \frac{OverallAcc - ExpAcc}{1 - ExpAcc}$$

$$ExpAcc = \frac{x_{+i}x_{i+}}{(\sum_{i,j=1}^{M} x_{ij})^2}$$

$$\sigma^2_{Kappa} = \frac{OverallAcc - ExpAcc}{(1 - ExpAcc)^2 \cdot N_{Total}}$$

where:

- $OverallAcc$: overall accuracy.
- $ExpAcc$: expected accuracy of agreement if agreement were purely random.

Returns: A list of real values containing with kappa coefficient, its variance and confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$Kappa()
```

`ConfMatrix$ModKappa()`**:** Public method that provides the overall modified kappa coefficient. The method also offers the variance and confidence interval. The references Stehman (1997) and Foody (1992) are followed for the calculations.

$$ModKappa = \frac{OverallAcc - \frac{1}{M}}{1 - \frac{1}{M}}$$

$$\sigma^2_{ModKappa} = \frac{OverallAcc \cdot (1 - OverallAcc)}{\left(1 - \frac{1}{M}\right)^2 \cdot N_{Total}}$$

where:

- $OverallAcc$: overall accuracy.

Usage:

```
ConfMatrix$ModKappa(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing modified coefficient kappa, its variance and its confidence interval.

Examples:

```
A <- matrix(c(317,61,2,35,23,120,4,29,0,0,60,0,0,0,0,8),nrow=4,ncol=4)
p <- ConfMatrix$new(A,Source="Foody 1992")
p$ModKappa()
```

`ConfMatrix$UserKappa_i()`**:** Public method derived by the kappa coefficient evaluated from the user's perspective, for a specific class i. The method also offers the variance and confidence interval. The reference Rosenfield and Fitzpatrick-Lins (1986) is followed for the calculations.

Usage:

```
ConfMatrix$UserKappa_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Details:

$$UserKappa_i = \frac{UserAcc_i - \frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}}}{1 - \frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}}}$$

$$\sigma^2_{UserKappa_i} = \frac{UserAcc_i \cdot (1 - UserAcc_i)}{\left(1 - \frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}}\right)^2 \cdot N_i}$$

where:

- $UserAcc_i$: user accuracy index for class i.

Returns: A list of real values containing the kappa coefficient (user's perspective), its variance and its confidence interval.

Examples:

```
A<-matrix(c(73,13,5,1,0,21,32,13,3,0,16,39,35, 29,13,3,5,7,28,48,1,0,2,3,17),
nrow=5,ncol=5)
p<-ConfMatrix$new(A,Source="Næsset 1996")
p$UserKappa_i(2)
```

`ConfMatrix$ModKappaUser_i()`**:** Public method, derived from the general modified kappa coefficient, which provides the modified kappa coefficient from the user's perspective and for a specific class i. Equitable probabilities of belonging to each class are assumed. The method also offers the variance and confidence interval. The references Stehman (1997) and Foody (1992) are followed for the calculations.

$$ModKappaUser_i = \frac{UserAcc_i - \frac{1}{M}}{1 - \frac{1}{M}}$$

$$\sigma^2_{ModKappaUser_i} = \frac{UserAcc_i \cdot (1 - UserAcc_i)}{\left(1 - \frac{1}{M}\right)^2 \cdot N_i}$$

where:

- $UserAcc_i$: user accuracy index for class i.

Usage:

```
ConfMatrix$ModKappaUser_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing the modified kappa coefficient from the user's perspective, its variance and confidence interval.

Examples:

```
A<-matrix(c(0,12,0,0,12,0,0,0,0,0,0,12,0,0,12,0),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$ModKappaUser_i(2)
```

`ConfMatrix$ProdKappa_i()`: Public method derived by the kappa coefficient evaluated from the producer's perspective, for a specific class i. The method also offers the variance and confidence interval. The reference Rosenfield and Fitzpatrick-Lins (1986) is followed for the calculations.

Usage:

```
ConfMatrix$ProdKappa_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Details:

$$ProdKappa_i = \frac{ProdAcc_i - \frac{x_{+i}}{\sum_{i,j=1}^{M} x_{ij}}}{1 - \frac{x_{+i}}{\sum_{i,j=1}^{M} x_{ij}}}$$

$$\sigma^2_{ProdKappa_i} = \frac{ProdAcc_i \cdot (1 - ProdAcc_i)}{\left(1 - \frac{x_{+i}}{\sum_{i,j=1}^{M} x_{ij}}\right)^2 \cdot N_j}$$

where:

- $ProdAcc_i$: producer accuracy index for class i.

Returns: A list of real values containing the coefficient kappa (producer's), its variance and its confidence interval.

Examples:

```
A <- matrix(c(73,13,5,1,0,21,32,13,3,0,16,39,35,29,13,3,5,7,28,48,1,0,2,3,17),
nrow=5,ncol=5)
p<-ConfMatrix$new(A,Source="Næsset 1996")
p$ProdKappa_i(2)
```

`ConfMatrix$ModKappaProd_i()`: Public method, derived from the general modified kappa coefficient, which provides the modified kappa coefficient from the producer's perspective and for a specific class i. Equitable probabilities of belonging to each class are assumed. The method also offers the variance and confidence interval. The references Stehman (1997) and Foody (1992) are followed for the calculations.

Usage:

```
ConfMatrix$ModKappaProd_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Details:

$$ModKappaProd_i = \frac{ProdAcc_i - \frac{1}{M}}{1 - \frac{1}{M}}$$

$$\sigma^2_{ModKappaProd_i} = \frac{ProdAcc_i \cdot (1 - ProdAcc_i)}{\left(1 - \frac{1}{M}\right)^2 \cdot N_j}$$

where:

- $ProdAcc_i$: producer accuracy index for class i.

Returns: A list of real values containing the modified kappa coefficient from the producer's perspective, its variance and confidence interval.

Examples:

```
A<-matrix(c(317,61,2,35,23,120,4,29,0,0,60,0,0,0,0,8),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Foody 1992")
p$ModKappaProd_i(2)
```

`ConfMatrix$DetailKappa()`**:** Public method that calculates the general Kappa agreement index, its standard deviation and the test statistic to test its significance. The delta method has been used to calculate the sample variance. The reference Congalton and Green (2008) is followed for the computations.

$$Kappa = \frac{OverallAcc - ExpAcc}{1 - ExpAcc}$$

$$ExpAcc = \frac{x_{+j} \cdot x_{i+}}{\sum_{(i,j=1}^{M} x_{ij})^2}$$

$$\sigma^2_{Kappa} = \frac{1}{N_{Total}} \left(\frac{\theta_1(1-\theta_1)}{(1-\theta_2)^2} + \frac{2(1-\theta_1)(2\theta_1\theta_2 - \theta_3)}{(1-\theta_2)^3} + \frac{(1-\theta_1)^2(\theta_4 - 4\theta_2^2)}{(1-\theta_2)^4} \right)$$

where

$$\theta_1 = OverallAcc = \sum_{i,j=1}^{M} \frac{x_{ii}}{x_{ij}}$$

$$\theta_2 = ExpAcc = \sum_{i=1}^{M} \left(\frac{x_{+i}}{\sum_{j=1}^{M} x_{ij}} \cdot \frac{x_{i+}}{\sum_{j=1}^{M} x_{ij}} \right)$$

$$\theta_3 = \sum_{i=1}^{M} \left(\frac{x_{ii}x_{+i}}{\sum_{j=1}^{M} x_{ij}} \cdot \frac{x_{ii}x_{i+}}{\sum_{j=1}^{M} x_{ij}} \right)$$

$$\theta_4 = \frac{1}{(\sum_{i,j=1}^{M} x_{ij})^3} \sum_{i,j=1}^{M} x_{ij}(x_{j+} + x_{+i})^2$$

$$Z = \frac{Kappa}{\sqrt{\sigma^2_{Kappa}}}$$

Where:

- $ExpAcc$: expected accuracy of agreement if agreement were purely random.
- $OverallAcc$: overall accuracy.
- $\theta_1, \theta_2, \theta_3, \theta_4$: real values.
- Z: the test statistic.

Usage:

```
ConfMatrix$DetailKappa()
```

Returns: A list of real values containing the kappa coefficient, its standard deviation, and the value of its test statistic.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$DetailKappa()
```

`ConfMatrix$DetailCondKappa()`**:** Public method that calculates the Kappa class agreement index (conditional Kappa) from the perspective of user (i) and producer (j) and its standard deviations. The reference Congalton and Green (2008) is followed for the computations.

$$CondKappa_{user} = \frac{\frac{x_{ii}}{x_{i+}} - x_{+j}}{1 - x_{+j}}$$

$$CondKappa_{producer} = \frac{\frac{x_{ii}}{x_{+j}} - x_{i+}}{1 - x_{i+}}$$

$$\sigma^2_{CondKappa_{producer}} = \frac{1}{N_{Total}} \cdot \frac{x_{+j} - x_{ii}}{x^3_{+j}(1 - x_{i+})^3} \cdot ((x_{+j} - x_{ii}) \cdot (x_{+j}x_{i+} - x_{ii}) + x_{ii}(1 - x_{+j} - x_{i+} + x_{ii}))$$

$\sigma^2_{CondKappa_{user}}$ is done in an analogous way by exchanging x_{i+} to x_{+j}.

Usage:

```
ConfMatrix$DetailCondKappa()
```

Returns: A list of real values containing conditional Kappa index of the user's and the producer's, and its corresponding standard deviation.

Examples:

```
A<-matrix(c(0.2361,0.0694,0.1389,0.0556,0.1667,0.0417,0.1111,0,0.1806),
ncol=3,nrow=3)
p<-ConfMatrix$new(A,Source="Czaplewski 1994")
p$DetailCondKappa ()
```

`ConfMatrix$DetailWKappa()`**:** Public method that calculates the general Kappa agreement index (weighted) and its standard deviation. The reference Fleiss et al. (1969); Næsset (1996) and Congalton and Green (2008) are followed for the computations.

Be $p_{ij} = \frac{x_{ij}}{\sum_{i,j}^{M} x_{ij}}$ for each element i, j for the matrix and $0 \leq w_{ij} \leq 1$ for $i \neq j$ and $w_{ii} = 1$ for $i = j$. If the elements of the weight are greater than 1, their value must be given as a percentage. Therefore, let:

$$p_o = \sum_{i,j=1}^{M} w_{ij} p_{ij}$$

be the weighted agreement, and

$$p_c = \sum_{i,j=1}^{M} w_{ij} p_{i+} p_{+j}$$

with p_{i+}, p_{+j} analogous to x_{i+}, x_{+j}.
Then, the weighted Kappa is defined by

$$Kappa_w = \frac{p_o - p_c}{1 - p_c}$$

The variance may be estimated by

$$\sigma^2_{Kappa_w} = \frac{1}{N_{Total}(1-p_c)^4} \left(\sum_{i,j=1}^{M} p_{ij}[w_{ij}(1-p_c) - (\overline{w}_{i+} + \overline{w}_{+j})(1-p_o)]^2 - (p_o p_c - 2p_c + p_o)^2 \right)$$

where $\overline{w}_{i+} = \sum_{j=1}^{M} w_{ij} p_{+j}$ and $\overline{w}_{+j} = \sum_{i=1}^{M} w_{ij} p_{i+}$
Its statistic is given by:

$$Z = \frac{Kappa_W}{\sqrt{\sigma^2_{Kappa_W}}}$$

Usage:

```
ConfMatrix$DetailWKappa(WM)
```

Arguments:

`WM` Weight matrix (as matrix).

Returns: A list with the weight matrix, kappa index obtained from the original matrix and the weight matrix, its standard deviations and the value of its test statistic.

Examples:

```
A <- matrix(c(1,1,0,0,0,5,55,27,23,0,3,30,68,74,4,0,8,8,39,26,0,0,2,4,26),
nrow=5)
WM <- matrix(c(1,0.75,0.5,0.25,0,0.75,1,0.75,0.5,0.25,0.5,0.75,1,0.75,0.5,
0.25,0.5,0.75,1,0.75,0,0.25,0.5,0.75,1),nrow=5)
p<-ConfMatrix$new(A, Source="Næsset 1996")
p$DetailWKappa(WM)
```

`ConfMatrix$Tau()`**:** Public method that calculates the Tau index and its variance. Its value indicates how much the classification has improved compared to a random classification of the N elements into M groups. The method also offers the variance and confidence interval. The reference Ma and Redmond (1995) is followed for the computations.

$$Tau = \frac{OverallAcc - PrAgCoef}{1 - PrAgCoef}$$

$$PrAgCoef = \frac{1}{M}$$

$$\sigma^2_{Tau} = \frac{OverallAcc \cdot (1 - OverallAcc)}{N_{Total} \cdot (1 - PrAgCoef)^2}$$

Where:

- $OverallAcc$: overall accuracy.
- $PrAgCoef$: a priori random agreement coefficient.

Usage:

```
ConfMatrix$Tau(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing the Tau index, its variance and confidence interval.

Examples:

```
A<-matrix(c(238051,7,132,0,0,24,9,2,189,1,4086,188,0,4,16,45,1,0,939,5082,
51817,0,34,500,1867,325,17,0,0,5,11148,1618,78,0,0,0,0,48,4,834,2853,340,
32,0,197,5,151,119,135,726,6774,75,1,553,0,105,601,110,174,155,8257,8,0,
29,36,280,0,0,6,5,2993,0,115,2,0,4,124,595,0,0,4374),nrow=9,ncol=9)
p<-ConfMatrix$new(A,Source="Muñoz 2016")
p$Tau()
```

`ConfMatrix$DetailWTau()`**:** Public method that calculates the general Tau concordance index (weighted) and its standard deviation.
Be $p_{ij} = \frac{x_{ij}}{\sum_{i,j}^{M} x_{ij}}$ for each element i, j for the matrix and $0 \leq w_{ij} \leq 1$ for $i \neq j$ and $w_{ii} = 1$ for $i = j$. If the elements of the weight are greater than 1, their value must be given as a percentage. The following real values are defined:

$$\theta_1 = \sum_{i}^{M} p_{ii}$$

$$\theta_2 = \sum_{i=1}^{M} w_{ij} p_{i+}$$

$$\theta_3 = \sum_{i=1}^{M} (p_{ii}(w_{ij} + p_{+j}))$$

$$\theta_4 = \sum_{i,j=1}^{M} p_{ij} m_{ij}$$

where m_{ij} are the elements of a matrix, which are given by $(w_{ij} + p_{+j})^2$
Therefore,

$$Tau_W = \frac{\theta_1 - \theta_2}{1 - \theta_2}$$

$$\sigma^2_{Tau_W} = \frac{1}{N_{Total}} \left(\frac{\theta_1(1-\theta_1)}{(1-\theta_2)^2} + 2\frac{1-\theta_1}{(1-\theta_2)^3}(2\theta_1\theta_2 - \theta_3) + \frac{(1-\theta_1)^2}{(1-\theta_2)^4}(\theta_4 - 4\theta_2^2) \right)$$

The statistic is given by

$$Z = \frac{Tau_W}{\sqrt{\sigma^2_{Tau_W}}}$$

Where:

- $\theta_1, \theta_2, \theta_3, \theta_4$: real values.
- Z: the test statistic.

Usage:

```
ConfMatrix$DetailWTau(WV)
```

Arguments:

```
WV
   Weights vector (as matrix)
```

Returns: A list with the weighted Tau index, the weight matrix, its standard deviation and its statistics.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
WV <-matrix(c(0.4, 0.1, 0.4, 0.1),ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$DetailWTau(WV)
```

`ConfMatrix$Ent()`**:** Public method for calculating product entropy,which refers to the lack of orden and predictability that the product presents. The method also offers the variance and confidence interval. The reference Finn (1993) is followed for the calculations.

$$Ent = \sum_{i,j=1}^{M} \left(\frac{x_{ij}}{\sum_{i,j=1}^{M} x_{ij}} \cdot \log \left(\frac{x_{ij}}{\frac{x_{i+} \cdot x_{+j}}{\sum_{i,j=1}^{M} x_{ij}}} \right) \right)$$

$$\sigma^2_{Ent} = \frac{Ent \cdot (1 - Ent)}{N_{Total}}$$

Usage:

```
ConfMatrix$Ent(alpha = NULL, v = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

`v`
Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. By default v=10(units Hartleys), v=2(units bits), v=e(units nats).

Returns: A list of real values containing the entropy, its variance and confidence interval.

Examples:

```
A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$Ent(v=2)
```

`ConfMatrix$AvNormEnt()`: Public method that calculates normalized entropy using the arithmetic mean of the entropies on the product and the reference. The method also offers the variance and confidence interval. The reference Strehl and Ghosh (2002) is followed for the calculations.

$$AvNormEnt = \frac{2Ent}{Ent_i(A) + Ent_i(B)}$$

$$Ent_i(A) = -\sum_{j=1}^{M} \left(\left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$Ent_i(B) = -\sum_{i=1}^{M} \left(\left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$\sigma^2_{AvNormEnt} = \frac{AvNormEnt \cdot (1 - AvNormEnt)}{N_{Total}}$$

where:

- Ent: product entropy.
- $Ent_i(A)$: entropy with respect to the classes *i* of the product. A is a matrix.
- $Ent_i(B)$: entropy with respect to the class *i* on the reference. B is a matrix.

Usage:

```
ConfMatrix$AvNormEnt(alpha = NULL, v = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

`v`
Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. By default v=10(units Hartleys), v=2(units bits), v=e(units nats).

Returns: A list of real values containing the normalized entropy (arithmetic mean of the entropies on the product and reference), its variance and confidence interval.

Examples:

```
A<-matrix(c(0,12,0,0,12,0,0,0,0,0,0,12,0,0,12,0),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$AvNormEnt(v=2)
```

`ConfMatrix$GeomAvNormEnt()`: Public method that calculates the normalized entropy using the geometric mean of the product and reference entropies. The method also offers the variance and confidence interval. The reference Ghosh et al. (2002) is followed for the calculations.

Usage:

```
ConfMatrix$GeomAvNormEnt(alpha = NULL, v = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

`v` Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. By default `v=10(units Hartleys), v=2(units bits), v=e(units nats).`

Details:

$$GeomAvNormEnt = \frac{Ent}{\sqrt{Ent_i(A) \cdot Ent_i(B)}}$$

$$Ent_i(A) = -\sum_{j=1}^{M} \left(\left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$Ent_i(B) = -\sum_{i=1}^{M} \left(\left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$\sigma^2_{GeomAvNormEnt} = \frac{GeomAvNormEnt \cdot (1 - GeomAvNormEnt)}{N_{Total}}$$

where:

- Ent: product entropy.
- $Ent_i(A)$: entropy with respect to the classes *i* of the product. A is a matrix.
- $Ent_i(B)$: entropy with respect to the class *i* of the reference. B is a matrix.

Returns: A list of real values containing the normalized entropy (geometric mean of the entropies on the product and reference), its variance and confidence interval.

Examples:

```
A<-matrix(c(0,12,0,0,12,0,0,0,0,0,0,12,0,0,12,0),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$GeomAvNormEnt(v=2)
```

`ConfMatrix$AvMaxNormEnt()`: Public method that provides normalized entropy using the arithmetic mean of the maximum entropies of the product and reference. The method also offers the variance and confidence interval. The reference Strehl (2002) is followed for the calculations.

Usage:

```
ConfMatrix$AvMaxNormEnt(alpha = NULL, v = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

v Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. By default v=10(units Hartleys), v=2(units bits), v=e(units nats).

Details:

$$AvMaxNormEnt = \frac{2Ent}{max(Ent_i(A)) + max(Ent_i(B))} = \frac{Ent}{\log M}$$

$$Ent_i(A) = -\sum_{j=1}^{M} \left(\left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$Ent_i(B) = -\sum_{i=1}^{M} \left(\left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$\sigma^2_{AvMaxNormEnt} = \frac{AvMaxNormEnt \cdot (1 - AvMaxNormEnt)}{N_{Total}}$$

where:

- Ent: product entropy.
- $Ent_i(A)$: entropy with respect to the classes *i* of the product. A is a matrix.
- $Ent_i(B)$: entropy with respect to the class *i* on the reference. B is a matrix.

Returns: A list of real values containing the normalized entropy (arithmetic mean of the maximum entropies of the product and of reference), its variance, and its confidence interval.

Examples:

```
A<-matrix(c(8,0,0,0,0,16,0,0,0,0,8,0,0,0,0,16),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$AvMaxNormEnt(v=2)
```

`ConfMatrix$EntUser_i()`: Public method that calculates relative change of entropy for a given class i of the product. The method also offers the variance and confidence interval. The reference Finn (1993) is followed for the calculations.

Usage:

```
ConfMatrix$EntUser_i(i, alpha = NULL, v = NULL)
```

Arguments:

i Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

alpha Significance level. By default 0.05.

v Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. By default v=10(units Hartleys), v=2(units bits), v=e(units nats).

Details:

$$EntUser_i = \frac{Ent_i(A) - Ent_i(A|b_i)}{Ent_i(A)}$$

$$Ent_i(A) = -\sum_{j=1}^{M} \left(\left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$Ent_i(A|b_i) = -\sum_{j=1}^{M} \left(\left(\frac{x_{ij}}{x_{i+}} \right) \cdot \log \left(\frac{x_{ij}}{x_{i+}} \right) \right)$$

$$\sigma^2_{EntUser_i} = \frac{EntUser_i \cdot (1 - EntUser_i)}{N_{Total}}$$

where:

- $Ent_i(A)$: entropy with respect to the classes *i* of the product. A is a matrix.
- $Ent_i(A|b_i)$: Producer entropy knowing that the location corresponding to reference B is in class b_i. B is a matrix.

Returns: A list of real values containing the relative change of entropy for given class i, its variance, its confidence interval, producer's entropy, and producer's entropy knowing that the location corresponding to reference B is in class b_i.

Examples:

```
A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$EntUser_i(1,v=2)
```

`ConfMatrix$NormEntUser()`**:** Public method that calculates normalized entropy of the product. The method also offers the variance and confidence interval. The reference Finn (1993) is followed for the calculations.

Usage:

```
ConfMatrix$NormEntUser(alpha = NULL, v = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

`v` Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. `By default v=10(units Hartleys), v=2(units bits), v=e(units nats).`

Details:

$$NormEntUser = \frac{Ent}{Ent_i(B)}$$

$$Ent_i(B) = -\sum_{i=1}^{M} \left(\left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$\sigma^2_{NormEntUser} = \frac{NormEntUser \cdot (1 - NormEntUser)}{N_{Total}}$$

where:

- Ent: product entropy.
- $Ent_i(B)$: entropy with respect to the class *i* on the reference. B is a matrix.

Returns: A list of real values containing with normalized entropy of the product class i, conditioned to reference data, its variance and confidence interval.

Examples:

```
A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$NormEntUser(v=2)
```

`ConfMatrix$EntProd_i()`: Public method that calculates relative change of entropy for a given a class i of the reference from the producer's perspective. The method also offers the variance and confidence interval. The reference Stehman (1997) is followed for the calculations.

$$EntProd_i = \frac{Ent_i(B) - Ent_i(B|a_j)}{Ent_i(B)}$$

$$Ent_i(B) = -\sum_{i=1}^{M} \left(\left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \cdot \log \left(\frac{x_{i+}}{\sum_{i,j=1}^{M} x_{ij}} \right) \right)$$

$$Ent_i(B|a_j) = -\sum_{i=1}^{M} \left(\left(\frac{x_{ij}}{x_{+j}} \right) \cdot \log \left(\frac{x_{ij}}{x_{+j}} \right) \right)$$

$$\sigma^2_{EntProd_i} = \frac{EntProd_i \cdot (1 - EntProd_i)}{N_{Total}}$$

where:

- $Ent_i(B)$: entropy with respect to the class *i* on the reference. B is a matrix.
- $Ent_i(B|a_j)$: Entropy of reference B knowing that the location of product A is in the class a_j.

Usage:

```
ConfMatrix$EntProd_i(i, alpha = NULL, v = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

`v` Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. By default v=10(units Hartleys), v=2(units bits), v=e(units nats).

Returns: A list of real values containing the relative change of entropy for given class i, its variance, its confidence interval, entropy with respect to reference classes, and entropy with respect to reference classes knowing that the location corresponding to A is in class a_j.

Examples:

```
A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$EntProd_i(3,v=2)
```

`ConfMatrix$NormEntProd()`: Public method that calculates normalized entropy of the reference from the producer's perspective. The method also offers the variance and confidence interval. The reference Finn (1993) is followed for the calculations.

Usage:

```
ConfMatrix$NormEntProd(alpha = NULL, v = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

`v` Base of the logarithm, where $v \in \mathbb{R}^+ - \{1\}$. By default v=10(units Hartleys), v=2(units bits), v=e(units nats).

Details:

$$NormEntProd = \frac{Ent}{Ent_i(A)}$$

$$Ent_i(A) = -\sum_{j=1}^{M}\left(\left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}}\right)\cdot \log\left(\frac{x_{+j}}{\sum_{i,j=1}^{M} x_{ij}}\right)\right)$$

$$\sigma^2_{NormEntProd} = \frac{NormEntProd\cdot(1-NormEntProd)}{N_{Total}}$$

where:

- Ent: product entropy.
- $Ent_i(A)$: entropy with respect to the classes *i* of the product. A is a matrix.

Returns: A list of real values containing the normalized entropy of the reference class i from the producer's perspective, its variance and confidence interval.

Examples:

```
A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$NormEntProd(v=2)
```

`ConfMatrix$Sucess()`**:** Public method that provides the Classification Success Index (CSI) which considers all classes and gives an overall estimation of classification effectiveness. The method also offers the variance and confidence interval. The references Koukoulas and Blackburn (2001) and Türk (2002) are followed for the calculations.

$$Sucess = 1-(1-AvUserAcc+1-AvProdAcc) = AvUserAcc+AvProdAcc-1$$

$$\sigma^2_{Sucess} = \frac{Sucess\cdot(1-Sucess)}{N_{Total}}$$

where:

- $AvUserAcc$: average accuracy from user's perspective.
- $AvProdAcc$: average accuracy from producer's perspective.

Usage:

```
ConfMatrix$Sucess(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing the ICSI, its variance and its confidence interval.

Examples:

```
A<-matrix(c(0.3,0.02,0.01,0.12,0.19,0.03,0.02,0.01,0.3),nrow=3,ncol=3)
p<-ConfMatrix$new(A,Source="Labatut and Cherifi 2011")
p$Sucess()
```

`ConfMatrix$Sucess_i()`**:** Public method that provides the Individual Classification Success Index (ICSI) which considers the classification effectiveness for one particular class of interest. The

method also offers the variance and confidence interval. The references Koukoulas and Blackburn (2001) and Türk (2002) are followed for the calculations.

$$Sucess_i = 1 - (1 - UserAcc_i + 1 - ProdAcc_i) = UserAcc_i + ProdAcc_i - 1$$

$$\sigma^2_{Sucess_i} = \frac{Sucess_i \cdot (1 - Sucess_i)}{N_{ij}}$$

where:

- $UserAcc_i$: user accuracy index for class i.
- $ProdAcc_i$: producer accuracy index for class i.

Usage:

```
ConfMatrix$Sucess_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing the ICSI, its variance and its confidence interval.

Examples:

```
A<-matrix(c(0.3,0.02,0.01,0.12,0.19,0.03,0.02,0.01,0.3),nrow=3,ncol=3)
p<-ConfMatrix$new(A,Source="Labatut and Cherifi 2011")
p$Sucess_i(2)
```

`ConfMatrix$AvHellAcc()`**:** Public method that provides the average value of the Hellden mean precision index. Denoted by the probability that a randomly chosen position or element assigned to a specific class on the product has a correspondence of the same class in the homologous position or element in the reference, and that a randomly chosen point or element assigned to a specific class on the reference has a correspondence of the same class in the homologous position or element in the product. The method also offers the variance and confidence interval. The reference Liu et al. (2007) is followed for the calculations.

Usage:

```
ConfMatrix$AvHellAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$AvHellAcc = \frac{1}{M} 2 \sum_{i=1}^{M} \frac{x_{ii}}{x_{+i} + x_{i+}}$$

$$\sigma^2_{AvHellAcc} = \frac{AvHellAcc \cdot (1 - AvHellAcc)}{N_{Total}}$$

Returns: A list of real values containing the average of Hellden's mean accuracy index, its variance and confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvHellAcc()
```

`ConfMatrix$AvHellAcc_i()`**:** Public method that provides the Hellden' average accuracy for the specified class. The method also offers the variance and confidence interval. The references Hellden (1980) and Rosenfield and Fitzpatrick-Lins (1986) are followed for the calculations.

Usage:

```
ConfMatrix$AvHellAcc_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Details:

$$AvHellAcc_i = \frac{2}{\frac{1}{UserAcc_i} + \frac{1}{ProdAcc_i}} = \frac{2UserAcc_i \cdot ProdAcc_i}{UserAcc_i + ProdAcc_i}$$

$$\sigma^2_{AvHellAcc_i} = \frac{AvHellAcc_i \cdot (1 - AvHellAcc_i)}{N_{ij}}$$

where:

- $UserAcc_i$: user accuracy index for class i.
- $ProdAcc_i$: producer accuracy index for class i.

Returns: A list of real values containing the Hellden's mean accuracy, its variance and its confidence interval.

Examples:

```
A <- matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Rosenfield and Fitzpatrick 1986")
p$AvHellAcc_i(2)
```

`ConfMatrix$AvShortAcc()`**:** Public method that provides the average of the Short's mapping accuracy index. It is determined as the quotient between the well-classified elements (value on the diagonal) and the subtraction of that same value on the errors of omission and commission (rest of values in the column and row) corresponding to each class. The method also offers the variance and confidence interval. The reference Liu et al. (2007) is followed for the calculations.

Usage:

```
ConfMatrix$AvShortAcc(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Details:

$$AvShortAcc = \frac{1}{M} \sum_{i=1}^{M} \frac{x_{ii}}{\overline{x}_{+i} + \overline{x}_{i+} - x_{ii}}$$

$$\sigma^2_{AvShortAcc} = \frac{AvShortAcc \cdot (1 - AvShortAcc)}{N_{Total}}$$

Returns: A list of real values containing the average of Short's mapping accuracy index, its variance and confidence interval.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvShortAcc()
```

`ConfMatrix$ShortAcc_i()`**:** Public method that provides the Short's mapping accuracy for each class. The method also offers the variance and confidence interval. The references Rosenfield and Fitzpatrick-Lins (1986) and Short (1982) are followed for the calculations.

$$ShortAcc_i = \frac{x_{ii}}{\overline{x}_{+i} + \overline{x}_{i+} - x_{ii}}$$

$$\sigma^2_{ShortAcc_i} = \frac{ShortAcc_i \cdot (1 - ShortAcc_i)}{N_{ij}}$$

Usage:

```
ConfMatrix$ShortAcc_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Returns: A list of real values containing the Short's mapping accuracy, its variance and its confidence interval.

Examples:

```
A <- matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Rosenfield and Fitzpatrick-Lins 1986")
p$ShortAcc_i(2)
```

`ConfMatrix$GroundTruth()`**:** Public method that calculates the Ground Truth index, its variance and confidence interval.The reference Türk (1979) is followed for the computations.

To calculate R we begin the following iterative process:

Be $U_j^{(0)} = f_j^0$ with $f_j^0 = \frac{\overline{x}_{i+}}{\sum_{i=1}^{M} \overline{x}_{i+}}$ and $f_i^0 = \frac{\overline{x}_{+i}}{\sum_{i=1}^{M} \overline{x}_{+i}}$

Where $2m$ with $m = 1, 2, \cdots$

$$V_{i,2m-1} = \frac{f_i^0}{U_{+,2m-2} - U_{i,2m-2}}$$

where $U_{+,2m} = \sum_{i=1}^{M} U_{j,2m}$ and when $2m + 1$ with $m = 1, 2, \cdots$

$$U_{j,2m} = \frac{f_j^0}{V_{+,2m-1} - V_{i,2m-1}}$$

where $V_{+,2m-1} = \sum_{i=1}^{M} V_{i,2m-1}$

The iterative steps continue for $m = 1, 2, \cdots$ until the accuracy stabilizes thus taking the V term. Where

$$R = \frac{V}{\sum_{i=1}^{M} V_i}$$

$$ProdAcc = \frac{x_{ii}}{\sum_{j=1}^{M} x_{+j}}$$

$$GroundTruth = \frac{ProdAcc - R}{1 - R}$$

$$\sigma^2_{GroundTruth} = \frac{GroundTruth \cdot (1 - GroundTruth)}{N_{Total}}$$

Where:

- R: casual lucky guess.
- $ProdAcc$: producer accuracy.

Usage:

```
ConfMatrix$GroundTruth(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default 0.05.

Returns: A list with Ground Truth indexes, their variance, confidence intervals and the matrix with the expected frequencies.

Examples:

```
A<-matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Türk 1979")
p$GroundTruth()
```

`ConfMatrix$GroundTruth_i()`**:** Public method that calculates the Ground Truth index for class i, its variance and confidence interval.The reference Türk (1979) is followed for the computations.

To calculate R_i we begin the following iterative process: Be $U_j^{(0)} = f_j^0$ with $f_j^0 = \frac{\overline{x}_{i+}}{\sum_{i=1}^{M} \overline{x}_{i+}}$ and $f_i^0 = \frac{\overline{x}_{+i}}{\sum_{i=1}^{M} \overline{x}_{+i}}$

Where $2m$ with $m = 1, 2, \cdots$

$$V_{i,2m-1} = \frac{f_i^0}{U_{+,2m-2} - U_{i,2m-2}}$$

where $U_{+,2m} = \sum_{i=1}^{M} U_{j,2m}$ and when $2m + 1$ with $m = 1, 2, \cdots$

$$U_{j,2m} = \frac{f_j^0}{V_{+,2m-1} - V_{i,2m-1}}$$

where $V_{+,2m-1} = \sum_{i=1}^{M} V_{i,2m-1}$

The iterative steps continue for $m = 1, 2, \cdots$ until the accuracy stabilizes thus taking the V term. Where

$$R_i = \frac{V_i}{\sum_{i=1}^{k} V_i}$$

$$ProdAcc_i = \frac{x_{ii}}{\sum_{j=1}^{M} x_{+j}}$$

$$GroundTruth_i = \frac{ProdAcc_i - R_i}{1 - R_i}$$

$$\sigma^2_{GroundTruth_i} = \frac{GroundTruth_i \cdot (1 - GroundTruth_i)}{N_{Total}}$$

Where:

- R_i: casual lucky guess for class i. Is a real value.
- $ProdAcc_i$: producer accuracy for class i.

Usage:

```
ConfMatrix$GroundTruth_i(i, alpha = NULL)
```

Arguments:

`i` Class to evaluate, where $i \in \mathbb{Z} - \{0\}$.

`alpha` Significance level. By default 0.05.

Returns: A list with Ground Truth index for class i, its variance, confidence interval and the matrix with the expected frequencies for all classes.

Examples:

```
A<-matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Türk 1979")
p$GroundTruth_i(3)
```

`ConfMatrix$HellingerDist()`**:** Public method that provides that Hellinger distance between two confusion matrices. The reference García-Balboa et al. (2018) is followed for the computations.

$$HellingerDist = \frac{4 n_A m_B}{n_A + m_B} \sum_{i=1}^{M} (\sqrt{p_i} - \sqrt{q_i})^2$$

Where:

- n_A: sum of elements of the matrix A.
- m_B: sum of elements of the matrix B.
- p_i: probability that element $i \in [1, \cdots, MxM]$ is well classified in matrix A.
- q_i: probability that element $i \in [1, \cdots, MxM]$ is well classified in matrix B.

Usage:

```
ConfMatrix$HellingerDist(f, p = NULL, q = NULL)
```

Arguments:

`f` Element of the ConfMatrix.

`p` probability vector of matrix A. By default, relative frequencies observed for each cell is taken.

`q` probability vector of matrix B. By default, relative frequencies observed for each cell is taken.

Returns: A real value for the Hellinger distance.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
r<-ConfMatrix$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
r$HellingerDist(f)
```

`ConfMatrix$QES()`**:** Public method that calculates the values of quantity difference, exchange and shift. Quantity difference is the amount of difference between the product and the reference and is due to the less than maximum match in the proportions of the categories. Exchange represents transitions from class i to j and a transition from class j to class i in an identical number of cases. Shift refers to the difference remaining after subtracting quantity difference and exchange from the overall difference. The reference Pontius Jr and Santacruz (2014) is followed for the computations.

Where

$$Q = \frac{\sum_{j=1}^{M} q_j}{2}$$

$$E = \frac{\sum_{j=1}^{M} e_j}{2}$$

$$S = \frac{\sum_{j=1}^{M} s_j}{2}$$

with

$$d_j = \frac{\left(\sum_{i=1}^{M}(x_{ij} + x_{ji})\right) - 2x_{jj}}{\sum_{i=1}^{M}\sum_{j=1}^{M} x_{ij}}$$

$$q_j = \frac{\left|\sum_{i=1}^{M}(x_{ij} + x_{ji})\right|}{\sum_{i=1}^{J}\sum_{j=1}^{J} x_{ij}}$$

$$e_j = \frac{2\left(\left(\sum_{i=1}^{M} min(x_{ij}, x_{ji})\right) - x_{jj}\right)}{\sum_{i=1}^{M}\sum_{j=1}^{M} x_{ij}}$$

$$s_j = d_j - q_j - e_j$$

Usage:

```
ConfMatrix$QES()
```

Returns: A list of integer values with quantity, exchange, and shift. In addition to the differences for classes of the components of quantity, exchange and turn.

Examples:

```
A<-matrix(c(3,2,1,1,3,3,2,0,1),nrow=3,ncol=3)
p<-ConfMatrix$new(A,Source="Pontius Jr. and Santacruz 2023")
p$QES()
```

`ConfMatrix$MTypify()`**:** Public method that typifies the confusion matrix. The total sum of the original matrix is used for typing. In a typed matrix the sum of all values is unity. The resulting values can be presented as real values (parameter RaR=1), or as a percentage (parameter RaR !=1).

$$MTypify = \frac{x_{ij}}{\sum_{i,j=1}^{M} x_{ij}}$$

Usage:

```
ConfMatrix$MTypify(RaR = NULL)
```

Arguments:

`RaR` "1" indicates result as real, other values mean percentage as integer. By default RaR=1.

Returns: A list with two arrays, the first is the original array, the second the typed one.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A, Source="Congalton and Green 2008")
p$MTypify(RaR=5)
```

`ConfMatrix$MBootStrap()`**:** Public method that provides B resamples, using a multinomial distribution, of the confusion matrix of a ConfMatrix object. As a result, a set of bootstrapped cases is offered. The reference Fienberg (1970) is followed for the computations.

Usage:

```
ConfMatrix$MBootStrap(B, pr = NULL)
```

Arguments:

`B` Number of resamples.

`pr` Vector with resampling probabilities. By default, the success probability of each cell will be taken.

Returns: A list of B + 1 arrays formed by the original confusion matrix and all the simulated cases.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A, Source="Congalton and Green 2008")
p$MBootStrap(2)
```

`ConfMatrix$MNormalize()`**:** Public method that carries out an iterative process in order to equals one the sum of values by rows and columns. The references Fienberg (1970) and Muñoz (2016) are followed for the computations.

The following iterative process is used:

Let x_{ij} be the elements of the instance. It defines:

$$x'_{ij} = \frac{x_{ij}}{x_{i+}}$$

$$x''_{ij} = \frac{x'_{ij}}{x'_{+j}}$$

Taking $x_{ij} = x''_{ij}$ for the next iteration.

Usage:

```
ConfMatrix$MNormalize(iter = NULL)
```

Arguments:

`iter` Number of iteration. By default iter=1000.

Returns: A list formed by the original confusion matrix and the normalized matrix.

Examples:

```
A<-matrix(c(238051,7,132,0,0,24,9,2,189,1,4086,188,0,4,16,45,1,0,939,5082,
51817,0,34,500,1867,325,17,0,0,5,11148,1618,78,0,0,0,0,48,4,834,2853,340,
32,0,197,5,151,119,135,726,6774,75,1,553,0,105,601,110,174,155,8257,8,0,
29,36,280,0,0,6,5,2993,0,115,2,0,4,124,595,0,0,4374),nrow=9,ncol=9)
p<-ConfMatrix$new(A,Source="Muñoz 2016")
p$MNormalize()
```

`ConfMatrix$MPseudoZeroes()`**:** Public method that small values are calculated for empty cells of the matrix. All non-empty cells of the matrix change their values. This function will not be applied if all the elements of the matrix are different from 0. The reference Muñoz (2016) is followed for the computations.

Let x_{ij} be the elements of the instance.

The following values are defined:

$$e_{ij} = \frac{x_{i+}x_{+j}}{\sum_{i,j=1}^{M} x_{ij}}$$

$$v = \frac{\left(\sum_{i,j=1}^{M} x_{ij}\right)^2 - \sum_{i,j=1}^{M} x_{ij}^2}{\sum_{i,j=1}^{M}(e_{ij} - x_{ij})^2}$$

$$p_{ij} = \frac{e_{ij} \cdot v}{\sum_{i,j=1}^{M} x_{ij}}$$

Finally, the elements of the pseudozero matrix Z will be given by:

$$z_{ij} = \left(\frac{\sum_{i,j=1}^{M} x_{ij}}{(\sum_{i,j=1}^{M} x_{ij}) + v}\right)(p_{ij} + x_{ij})$$

Usage:

```
ConfMatrix$MPseudoZeroes()
```

Returns: A list formed by the original confusion matrix and the Pseudozeroes matrix.

Examples:

```
A<-matrix(c(238051,7,132,0,0,24,9,2,189,1,4086,188,0,4,16,45,1,0,939,5082,
51817,0,34,500,1867,325,17,0,0,5,11148,1618,78,0,0,0,0,48,4,834,2853,340,
32,0,197,5,151,119,135,726,6774,75,1,553,0,105,601,110,174,155,8257,8,0,
29,36,280,0,0,6,5,2993,0,115,2,0,4,124,595,0,0,4374),nrow=9,ncol=9)
p<-ConfMatrix$new(A,Source="Muñoz 2016")
p$MPseudoZeroes()
```

`ConfMatrix$OverallAcc.test()`: Public method that tests whether two independent confusion matrices (instances of the ConfMatrix class), are significantly different using their overall accuracy indexes. The reference Congalton and Green (2008) and Ma and Redmond (1995) are followed for the computations.

$$Z = \frac{|O_A - O_B|}{\sqrt{(\sigma^2_{O_A} + \sigma^2_{O_B})}}$$

Where:

- O_A: overall index of matrix A.
- O_B: overall index of matrix B.
- $\sigma^2_{O_A}$: variance of O_A.
- $\sigma^2_{O_B}$: variance of O_B.

Usage:

```
ConfMatrix$OverallAcc.test(f)
```

Arguments:

`f` Instance of ConfMatrix class.

Returns: A list of class "htest" containing the results of the hypothesis test.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
p$OverallAcc.test(f)
```

`ConfMatrix$Kappa.test()`: Public method that tests whether two independent confusion matrices (instances of the ConfMatrix class), are significantly different when using the kappa indexes. The reference Congalton and Green (2008) is followed for the computations.

$$Z = \frac{|k_A - k_B|}{\sqrt{(\sigma^2_{k_A} + \sigma^2_{k_B})}}$$

Where:

- k_A: kappa index of matrix A.
- k_B: kappa index of matrix B.
- $\sigma^2_{k_A}$: variance of k_A.
- $\sigma^2_{k_B}$: variance of k_B.

Usage:

`ConfMatrix$Kappa.test(f)`

Arguments:

`f` Element of the ConfMatrix class.

Returns: A list of class "htest" containing the results of the hypothesis test.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
p$Kappa.test(f)
```

`ConfMatrix$Tau.test()`: Public method that tests whether two independent confusion matrices (instances of the ConfMatrix class), are significantly different using their Tau indexes. The reference Congalton and Green (2008) and Ma and Redmond (1995) are followed for the computations.

$$Z = \frac{|\tau_A - \tau_B|}{\sqrt{(\sigma^2_{\tau_A} + \sigma^2_{\tau_B})}}$$

Where:

- τ_A: Tau index of matrix A.
- τ_B: Tau index of matrix B.
- $\sigma^2_{\tau_A}$: variance of τ_A.
- $\sigma^2_{\tau_B}$: variance of τ_B.

Usage:

`ConfMatrix$Tau.test(f)`

Arguments:

`f` Element of the ConfMatrix class.

Returns: A list of class "htest" containing the results of the hypothesis test.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
p$Tau.test(f)
```

`ConfMatrix$TSCM.test()`: Public method that performs a homogeneity test based on the Hellinger distance between two confusion matrices (instances of the ConfMatrix class). The test considers the individual cell values in the matrices. Bootstrap is applied to the matrices to obtain a consistent estimator. The reference García-Balboa et al. (2018) are followed for the computations. The calculation consists of obtaining a statistic, which we will call $T_{n,m}$, between both matrices from ConfMatrix$HellingerDist. Bootstrap is then applied to the confusion matrices to obtain simulations of both matrices. ConfMatrix$HellingerDist is applied again to these simulations and we will obtain the statistics $T^*_{n,m}$. The p value is defined as:

$$\hat{p} = \frac{Card(T^*_{n,m} \geq T^*_{n,m})}{B}$$

Usage:

```
ConfMatrix$TSCM.test(f, B = NULL)
```

Arguments:

`f` Element of the ConfMatrix class.

`B` Number of bootstraps that you want to generate. By default B=1000.

Returns: A list of class "htest" containing the results of the hypothesis test.

Examples:

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
C<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(C,Source="Congalton and Green 2008")
p$TSCM.test(f)
```

`ConfMatrix$QIndep.test()`**:** Public method that performs the quasi-independence test for the elements of a confusion matrix. The reference Türk (1979) and Goodman (1968) are followed for the computations.

$$G^2 = 2 \cdot \sum \log \frac{x_{ij}}{E_{ij}}$$

Following the procedure for calculating the elements of the function ConfMatrix$GroundTruth, we will have to E_{ij} is obtained from:

$$f_{ij} = U_j \cdot V_i$$

$$f_{ij}^0 = f_{ij} - f_{ii}$$

$$M^0 = x_{ij} - x_{ii}$$

where the elements of M^0 are m_{ij}^0

$$E_{ij} = f_{ij}^0 \sum_{i,j=1}^{M} m_{ij}^0$$

Where:

- x_{ij}: matrix element. Observed frequency.
- E_{ij}: expected frequency.

Usage:

```
ConfMatrix$QIndep.test()
```

Returns: A list of class "htest" containing the results of the hypothesis test.

Examples:

```
A<-matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source= "Türk 1979")
p$QIndep.test()
```

References

Alba-Fernández MV, Ariza-López FJ, Rodríguez-Avi J, García-Balboa JL (2020). "Statistical methods for thematic-accuracy quality control based on an accurate reference sample." *Remote Sensing*, **12**(5), 816.

Ariza FJ, Pinilla C, Garcia JL (2011). "Comparación de matrices de confusión celda a celda mediante bootstraping."

Ariza-Lopez F, Rodríguez-Avi J, García-Balboa J, Mesas-Carrascosa F (2013). *FUNDAMENTOS DE EVALUACIÓN DE LA CALIDAD DE LA INFORMACIÓN GEOGRÁFICA*. Universidad de Jaén. Servicio de publicaciones. ISBN 978-84-8439-813-4.

Cohen J (1960). "A coefficient of agreement for nominal scales." *Educational and psychological measurement*, **20**(1), 37–46.

Congalton RG, Green K (2008). *Assessing the accuracy of remotely sensed data: principles and practices*. CRC press.

Czaplewski RL (1994). *Variance approximations for assessments of classification accuracy*, volume 316. US Department of Agriculture, Forest Service, Rocky Mountain Forest and Range Experiment Station.

Fienberg SE (1970). "An iterative procedure for estimation in contingency tables." *The Annals of Mathematical Statistics*, **41**(3), 907–917.

Finn JT (1993). "Use of the average mutual information index in evaluating classification error and consistency." *International Journal of Geographical Information Science*, **7**(4), 349–366.

Fleiss JL, Cohen J, Everitt BS (1969). "Large sample standard errors of kappa and weighted kappa." *Psychological bulletin*, **72**(5), 323.

Foody GM (1992). "On the compensation for chance agreement in image classification accuracy assessment." *Photogrammetric engineering and remote sensing*, **58**(10), 1459–1460.

García-Balboa JL, Alba-Fernández MV, Ariza-López FJ, Rodríguez-Avi J (2018). "Analysis of thematic similarity using confusion matrices." *ISPRS international journal of geo-information*, **7**(6), 233.

Ghosh J, Strehl A, Merugu S (2002). "A consensus framework for integrating distributed clusterings under limited knowledge sharing." In *Proc. NSF Workshop on Next Generation Data Mining*, 99–108.

Goodman LA (1968). "The analysis of cross-classified data: Independence, quasi-independence, and interactions in contingency tables with or without missing entries: Ra Fisher memorial lecture." *Journal of the American Statistical Association*, **63**(324), 1091–1131.

Hellden U (1980). "A test of landsat-2 imagery and digital data for thematic mapping illustrated by an environmental study in northern Kenya, Lund University." *Natural Geography Institute Report No. 47*.

Koukoulas S, Blackburn GA (2001). "Introducing new indices for accuracy evaluation of classified images representing semi-natural woodland environments." *Photogrammetric Engineering and Remote Sensing*, **67**(4), 499–510.

Labatut V, Cherifi H (2011). "Evaluation of performance measures for classifiers comparison." *arXiv preprint arXiv:1112.4133*.

Liu C, Frazier P, Kumar L (2007). "Comparative assessment of the measures of thematic classification accuracy." *Remote sensing of environment*, **107**(4), 606–616.

Ma Z, Redmond RL (1995). "Tau coefficients for accuracy assessment of classification of remote sensing data." *Photogrammetric Engineering and Remote Sensing*, **61**, 435-439.

Muñoz JMS (2016). "Análisis de Calidad Cartográfica mediante el estudio de la Matriz de Confusión." *Pensamiento matemático*, **6**(2), 9–26.

Næsset E (1996). "Use of the weighted Kappa coefficient in classification error assessment of thematic maps." *International Journal of Geographical Information Systems*, **10**(5), 591–603.

Pontius Jr RG, Santacruz A (2014). "Quantity, exchange, and shift components of difference in a square contingency table." *International Journal of Remote Sensing*, **35**(21), 7543–7554.

Pontius Jr. RG, Santacruz A (2023). *diffeR: Metrics of Difference for Comparing Pairs of Maps or Pairs of Variables*. R package version 0.0-8, https://CRAN.R-project.org/package=diffeR.

Rosenfield GH, Fitzpatrick-Lins K (1986). "A coefficient of agreement as a measure of thematic classification accuracy." *Photogrammetric engineering and remote sensing*, **52**(2), 223–227.

Short NM (1982). *The Landsat tutorial workbook: Basics of satellite remote sensing*, volume 1078. National Aeronautics and Space Administration, Scientific and Technical Information Branch.

Strehl A (2002). *Relationship-based clustering and cluster ensembles for high-dimensional data mining*. The University of Texas at Austin.

Strehl A, Ghosh J (2002). "Cluster ensembles—a knowledge reuse framework for combining multiple partitions." *Journal of machine learning research*, **3**(Dec), 583–617.

Tung F, LeDrew E (1988). "The determination of optimal threshold levels for change detection using various accuracy indexes." *Photogrammetric Engineering and Remote Sensing*, **54**(10), 1449–1454.

Türk G (1979). "Gt index: A measure of the success of prediction." *Remote Sensing of Environment*, **8**(1), 65–75.

Türk G (2002). "Map evaluation and chance correction." *Photogrammetric Engineering and Remote Sensing*, **68**, 123–125;133.

Examples

```
## ------------------------------------------------
## Method `ConfMatrix$new()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
cm<-ConfMatrix$new (A,ID="5",Date="27-10-2023",Source="Congalton and Green,
2008")


## ------------------------------------------------
## Method `ConfMatrix$plot.index()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$plot.index()

## ------------------------------------------------
## Method `ConfMatrix$plot.UserProdAcc()`
```

```
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$plot.UserProdAcc()


## ------------------------------------------------
## Method `ConfMatrix$print()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,ClassNames=c("Deciduous","conifer","agriculture",
"shrub"),Source="Congalton and Green 2008")
p$print()


## ------------------------------------------------
## Method `ConfMatrix$AllParameters()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AllParameters()


## ------------------------------------------------
## Method `ConfMatrix$UserAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$UserAcc()


## ------------------------------------------------
## Method `ConfMatrix$UserAcc_i()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90), nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$UserAcc_i(2)


## ------------------------------------------------
## Method `ConfMatrix$AvUserAcc()`
## ------------------------------------------------

A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$AvUserAcc()
```

```
## ------------------------------------------------
## Method `ConfMatrix$CombUserAcc()`
## ------------------------------------------------

A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$CombUserAcc()


## ------------------------------------------------
## Method `ConfMatrix$ProdAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$ProdAcc()


## ------------------------------------------------
## Method `ConfMatrix$ProdAcc_i()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$ProdAcc_i(1)


## ------------------------------------------------
## Method `ConfMatrix$AvProdAcc()`
## ------------------------------------------------

A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$AvProdAcc()


## ------------------------------------------------
## Method `ConfMatrix$CombProdAcc()`
## ------------------------------------------------

A<-matrix(c(352,43,89,203),nrow=2,ncol=2)
p<-ConfMatrix$new(A,Source="Tung and LeDrew 1988")
p$CombProdAcc()


## ------------------------------------------------
## Method `ConfMatrix$UserProdAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$UserProdAcc()
```

```
## ------------------------------------------------
## Method `ConfMatrix$CombUserProdAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$CombUserProdAcc()


## ------------------------------------------------
## Method `ConfMatrix$AvUserProdAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc()


## ------------------------------------------------
## Method `ConfMatrix$AvUserProdAcc_i()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc_i(2)


## ------------------------------------------------
## Method `ConfMatrix$UserProdAcc_W()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
WM<- t(matrix(c(1,0,0.67,1,0,1,0,0,1,0,1,1,0.91,0,0.61,1),nrow=4,ncol=4))
p$UserProdAcc_W(WM)


## ------------------------------------------------
## Method `ConfMatrix$OverallAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A)
p$OverallAcc()


## ------------------------------------------------
## Method `ConfMatrix$Kappa()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
```

```
p$Kappa()


## ------------------------------------------------
## Method `ConfMatrix$ModKappa()`
## ------------------------------------------------

A <- matrix(c(317,61,2,35,23,120,4,29,0,0,60,0,0,0,0,8),nrow=4,ncol=4)
p <- ConfMatrix$new(A,Source="Foody 1992")
p$ModKappa()


## ------------------------------------------------
## Method `ConfMatrix$UserKappa_i()`
## ------------------------------------------------

A<-matrix(c(73,13,5,1,0,21,32,13,3,0,16,39,35, 29,13,3,5,7,28,48,1,0,2,3,17),
nrow=5,ncol=5)
p<-ConfMatrix$new(A,Source="Næsset 1996")
p$UserKappa_i(2)


## ------------------------------------------------
## Method `ConfMatrix$ModKappaUser_i()`
## ------------------------------------------------

A<-matrix(c(0,12,0,0,12,0,0,0,0,0,0,12,0,0,12,0),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$ModKappaUser_i(2)


## ------------------------------------------------
## Method `ConfMatrix$ProdKappa_i()`
## ------------------------------------------------

A <- matrix(c(73,13,5,1,0,21,32,13,3,0,16,39,35,29,13,3,5,7,28,48,1,0,2,3,17),
nrow=5,ncol=5)
p<-ConfMatrix$new(A,Source="Næsset 1996")
p$ProdKappa_i(2)


## ------------------------------------------------
## Method `ConfMatrix$ModKappaProd_i()`
## ------------------------------------------------

A<-matrix(c(317,61,2,35,23,120,4,29,0,0,60,0,0,0,0,8),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Foody 1992")
p$ModKappaProd_i(2)


## ------------------------------------------------
## Method `ConfMatrix$DetailKappa()`
## ------------------------------------------------
```

```
A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$DetailKappa()


## ------------------------------------------------
## Method `ConfMatrix$DetailCondKappa()`
## ------------------------------------------------

A<-matrix(c(0.2361,0.0694,0.1389,0.0556,0.1667,0.0417,0.1111,0,0.1806),
ncol=3,nrow=3)
p<-ConfMatrix$new(A,Source="Czaplewski 1994")
p$DetailCondKappa ()


## ------------------------------------------------
## Method `ConfMatrix$DetailWKappa()`
## ------------------------------------------------

A <- matrix(c(1,1,0,0,0,5,55,27,23,0,3,30,68,74,4,0,8,8,39,26,0,0,2,4,26),
nrow=5)
WM <- matrix(c(1,0.75,0.5,0.25,0,0.75,1,0.75,0.5,0.25,0.5,0.75,1,0.75,0.5,
0.25,0.5,0.75,1,0.75,0,0.25,0.5,0.75,1),nrow=5)
p<-ConfMatrix$new(A, Source="Næsset 1996")
p$DetailWKappa(WM)


## ------------------------------------------------
## Method `ConfMatrix$Tau()`
## ------------------------------------------------

A<-matrix(c(238051,7,132,0,0,24,9,2,189,1,4086,188,0,4,16,45,1,0,939,5082,
51817,0,34,500,1867,325,17,0,0,5,11148,1618,78,0,0,0,0,48,4,834,2853,340,
32,0,197,5,151,119,135,726,6774,75,1,553,0,105,601,110,174,155,8257,8,0,
29,36,280,0,0,6,5,2993,0,115,2,0,4,124,595,0,0,4374),nrow=9,ncol=9)
p<-ConfMatrix$new(A,Source="Muñoz 2016")
p$Tau()


## ------------------------------------------------
## Method `ConfMatrix$DetailWTau()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
WV <-matrix(c(0.4, 0.1, 0.4, 0.1),ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$DetailWTau(WV)


## ------------------------------------------------
## Method `ConfMatrix$Ent()`
## ------------------------------------------------
```

```
A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$Ent(v=2)


## ------------------------------------------------
## Method `ConfMatrix$AvNormEnt()`
## ------------------------------------------------

A<-matrix(c(0,12,0,0,12,0,0,0,0,0,0,12,0,0,12,0),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$AvNormEnt(v=2)


## ------------------------------------------------
## Method `ConfMatrix$GeomAvNormEnt()`
## ------------------------------------------------

A<-matrix(c(0,12,0,0,12,0,0,0,0,0,0,12,0,0,12,0),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$GeomAvNormEnt(v=2)


## ------------------------------------------------
## Method `ConfMatrix$AvMaxNormEnt()`
## ------------------------------------------------

A<-matrix(c(8,0,0,0,0,16,0,0,0,0,8,0,0,0,0,16),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Liu et al. 2007")
p$AvMaxNormEnt(v=2)


## ------------------------------------------------
## Method `ConfMatrix$EntUser_i()`
## ------------------------------------------------

A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$EntUser_i(1,v=2)


## ------------------------------------------------
## Method `ConfMatrix$NormEntUser()`
## ------------------------------------------------

A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$NormEntUser(v=2)


## ------------------------------------------------
## Method `ConfMatrix$EntProd_i()`
```

```
## ------------------------------------------------

A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$EntProd_i(3,v=2)


## ------------------------------------------------
## Method `ConfMatrix$NormEntProd()`
## ------------------------------------------------

A<-matrix(c(35,4,12,2,14,11,9,5,11,3,38,12,1,0,4,2),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Finn 1993")
p$NormEntProd(v=2)


## ------------------------------------------------
## Method `ConfMatrix$Sucess()`
## ------------------------------------------------

A<-matrix(c(0.3,0.02,0.01,0.12,0.19,0.03,0.02,0.01,0.3),nrow=3,ncol=3)
p<-ConfMatrix$new(A,Source="Labatut and Cherifi 2011")
p$Sucess()


## ------------------------------------------------
## Method `ConfMatrix$Sucess_i()`
## ------------------------------------------------

A<-matrix(c(0.3,0.02,0.01,0.12,0.19,0.03,0.02,0.01,0.3),nrow=3,ncol=3)
p<-ConfMatrix$new(A,Source="Labatut and Cherifi 2011")
p$Sucess_i(2)


## ------------------------------------------------
## Method `ConfMatrix$AvHellAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvHellAcc()


## ------------------------------------------------
## Method `ConfMatrix$AvHellAcc_i()`
## ------------------------------------------------

A <- matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Rosenfield and Fitzpatrick 1986")
p$AvHellAcc_i(2)
```

```
## ------------------------------------------------
## Method `ConfMatrix$AvShortAcc()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
p$AvShortAcc()


## ------------------------------------------------
## Method `ConfMatrix$ShortAcc_i()`
## ------------------------------------------------

A <- matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Rosenfield and Fitzpatrick-Lins 1986")
p$ShortAcc_i(2)


## ------------------------------------------------
## Method `ConfMatrix$GroundTruth()`
## ------------------------------------------------

A<-matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Türk 1979")
p$GroundTruth()


## ------------------------------------------------
## Method `ConfMatrix$GroundTruth_i()`
## ------------------------------------------------

A<-matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source="Türk 1979")
p$GroundTruth_i(3)


## ------------------------------------------------
## Method `ConfMatrix$HellingerDist()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
r<-ConfMatrix$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
r$HellingerDist(f)


## ------------------------------------------------
## Method `ConfMatrix$QES()`
## ------------------------------------------------
```

```
A<-matrix(c(3,2,1,1,3,3,2,0,1),nrow=3,ncol=3)
p<-ConfMatrix$new(A,Source="Pontius Jr. and Santacruz 2023")
p$QES()


## ------------------------------------------------
## Method `ConfMatrix$MTypify()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A, Source="Congalton and Green 2008")
p$MTypify(RaR=5)

## ------------------------------------------------
## Method `ConfMatrix$MBootStrap()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A, Source="Congalton and Green 2008")
p$MBootStrap(2)


## ------------------------------------------------
## Method `ConfMatrix$MNormalize()`
## ------------------------------------------------

A<-matrix(c(238051,7,132,0,0,24,9,2,189,1,4086,188,0,4,16,45,1,0,939,5082,
51817,0,34,500,1867,325,17,0,0,5,11148,1618,78,0,0,0,0,48,4,834,2853,340,
32,0,197,5,151,119,135,726,6774,75,1,553,0,105,601,110,174,155,8257,8,0,
29,36,280,0,0,6,5,2993,0,115,2,0,4,124,595,0,0,4374),nrow=9,ncol=9)
p<-ConfMatrix$new(A,Source="Muñoz 2016")
p$MNormalize()


## ------------------------------------------------
## Method `ConfMatrix$MPseudoZeroes()`
## ------------------------------------------------

A<-matrix(c(238051,7,132,0,0,24,9,2,189,1,4086,188,0,4,16,45,1,0,939,5082,
51817,0,34,500,1867,325,17,0,0,5,11148,1618,78,0,0,0,0,48,4,834,2853,340,
32,0,197,5,151,119,135,726,6774,75,1,553,0,105,601,110,174,155,8257,8,0,
29,36,280,0,0,6,5,2993,0,115,2,0,4,124,595,0,0,4374),nrow=9,ncol=9)
p<-ConfMatrix$new(A,Source="Muñoz 2016")
p$MPseudoZeroes()


## ------------------------------------------------
## Method `ConfMatrix$OverallAcc.test()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
```

```
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
p$OverallAcc.test(f)


## ------------------------------------------------
## Method `ConfMatrix$Kappa.test()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
p$Kappa.test(f)


## ------------------------------------------------
## Method `ConfMatrix$Tau.test()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(B,Source="Congalton and Green 2008")
p$Tau.test(f)


## ------------------------------------------------
## Method `ConfMatrix$TSCM.test()`
## ------------------------------------------------

A<-matrix(c(65,6,0,4,4,81,11,7,22,5,85,3,24,8,19,90),nrow=4,ncol=4)
p<-ConfMatrix$new(A,Source="Congalton and Green 2008")
C<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),nrow=4,ncol=4)
f<-ConfMatrix$new(C,Source="Congalton and Green 2008")
p$TSCM.test(f)


## ------------------------------------------------
## Method `ConfMatrix$QIndep.test()`
## ------------------------------------------------

A<-matrix(c(148,1,8,2,0,0,50,15,3,0,1,6,39,7,1,1,0,6,25,1,1,0,0,1,6),nrow=5,
ncol=5)
p<-ConfMatrix$new(A,Source= "Türk 1979")
p$QIndep.test()
```

QCCS *Quality Control Columns Set Class*

Description

The difference between a QCCS and a confusion matrix is that while forming a confusion matrix requires that the reference and the product be more or less equivalent, for the QCCS it is required that the reference be actually of higher quality than the product. This forces us to leave the marginals corresponding to the reference fixed. That is why we work by columns. In this way, the QCCS class works with a confusion matrix expressed as a set of column vectors and it will be analyzed by columns. A QCCS is constructed by comparing a sample of a set of common positions in the product and the ground truth. Appropriate sampling methods must be applied to generate the QCCS. It is considered that the classes of the ground truth correspond to the columns and that the classes of the product to be valued correspond to the rows. On the other hand, the concept of QCCS is directly linked to quality control, so the specifications of this control must be indicated (Ariza-López et al. 2019). Specifications are stated as percentages. E.g. for class "A" under consideration, a minimum quality value is established (e.g. better than 90%), and maximum values of confusion with other categories (e.g. confusion between A and B less than 5%). The specifications are proportions of a multinomial. First, an object of this class of object must be created (instantiated) and then the methods that offer the index calculations will be invoked.

Error Messages

List of possible errors:

- `Error type 1`: Different number of data vectors and probability.
- `Error type 2`: Different number of elements in the pair of data vectors and probabilities.
- `Error type 3`: The sum of the elements of the data vectors is 0.
- `Error type 4`: The sum of each probability vectors must be 1.
- `Error type 5`: Some element of the data vector is negative.
- `Error type 6`: Some element of the probability vector is negative.

Public fields

`Vectors list`. Integer data for the column vectors.

`Prob list`. Probability specifications for each of the vectors.

`ID character`. Identifier (maximum 50 characters).

`Date Date`. Date of the quality control.

`ClassNames character`. Name of the classes (maximum 20 characters).

`Source character | NULL`.

Methods

Public methods:

- `QCCS$new()`
- `QCCS$print()`
- `QCCS$Exact.test()`
- `QCCS$Ji.test()`
- `QCCS$JiGlobal.test()`

`QCCS$new()`: Public method to create an instance of the QCCS class. At the time of creation, column set data and specification values must be provided. The same number of data and as specification values must be entered, and the pairs of data-specifications vectors must have the same size, otherwise an error will be provided. The optional possibility of adding metadata to the matrix is offered. The values of the data vectors represent the classes of ground truth.

Usage:

```
QCCS$new(
  Vectors,
  Prob,
  ID = NULL,
  Date = NULL,
  ClassNames = NULL,
  Source = NULL
)
```

Arguments:

`Vectors` List of integer values data for the vectors.

`Prob` List of probability values corresponding to each of the vectors.

`ID` Unique identifier (Optional). It is a character string with a maximum length of 50 characters. By default, QCCS_i will be taken as identification. Where i will be the number of QCCS instances already defined in the session.

`Date` Date of the quality control (Optional). Formatted as "YYYY-MM-DD". By defaults `Sys.Date()`.

`ClassNames` Name of the classes (Optional). It is given by a character strings vector whose elements are the name of the classes. Each element of the vector is a string of maximum 20 characters. By default for the column elements they will be PC_i (Producer class).

`Source` (Optional) Indicates where the "vectors" and "prob" parameters come from (article, proj- ect, etc.). It is suggested to enter a reference or a DOI. A character string with a maximum length of 80 characters can be entered. By default, is NULL.

Examples:

```
Vectors<-list(c(47,4,0),c(44,5,3))
Prob<-list(c(0.95,0.04,0.01),c(0.88,0.1,0.02))
A<-QCCS$new(Vectors,Prob,
Source="Ariza-Lopez et al. 2019")
```

`QCCS$print()`: Public method that shows all the data entered by the user.

Usage:

```
QCCS$print()
```

Returns: QCCS object identifier, Date, name of classes, source of data and data vectors and probability.

Examples:

```
Vectors<-list(c(18,0,3,0),c(27,19))
Prob<-list(c(0.85,0.1,0.03,0.02),c(0.8,0.2))
A<-QCCS$new(Vectors,Prob,
Source="Alba-Fernández et al. 2020")
A$print()
```

`QCCS$Exact.test()`: Public method that using a QCCS object instance calculates whether the data meets specifications. An exact test is applied to each of the multinomials that are defined for each column. The Bonferroni method is used. The references (Ariza-López et al. 2019) and (Alba-Fernández et al. 2020) are followed for the computations.

Usage:

```
QCCS$Exact.test(alpha = NULL)
```

Arguments:

`alpha` Significance level numeric (Optional). By default a=0.05.

Returns: A list of the `htest` class containing the results of the hypothesis test. The p-value returned is the lowest of those obtained for the data analyzed. In addition, the Bonferroni criterion value, the p-values obtained for each column, the original data vectors and the probability vectors are also returned as parameters of the `htest` class.

Examples:

```
Vectors<-list(c(47,4,0),c(40,5,3))
Prob<-list(c(0.95,0.04,0.01),c(0.88,0.1,0.02))
A<-QCCS$new(Vectors,Prob,
Source="Ariza-Lopez et al. 2019")
A$Exact.test()
```

`QCCS$Ji.test()`: Public method that using a QCCS object instance calculates whether the data meets specifications in each of the classes. The Chi square test is used. The Bonferroni method is used. The references (Ariza-López et al. 2019) and (Alba-Fernández et al. 2020) are followed for the computations.

Usage:

```
QCCS$Ji.test(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default a=0.05.

Returns: A list of the `htest` class containing the results of the hypothesis test. The p-value returned is the lowest of those obtained for the data analyzed. In addition, the Bonferroni criterion value, the obtained p-values, the degrees of freedom and the statistics obtained for each column, the original data vectors and the probability vectors are also returned as parameters of the `htest` class.

Examples:

```
Vectors<-list(c(18,0,3,0),c(27,19))
Prob<-list(c(0.85,0.1,0.03,0.02),c(0.8,0.2))
A <- QCCS$new(Vectors,Prob,
Source="Alba-Fernández et al. 2020")
A$Ji.test()
```

`QCCS$JiGlobal.test()`: Public method that using a QCCS object instance calculates whether the data meets specifications. The Chi square test is used. The references (Ariza-López et al. 2019) and (Alba-Fernández et al. 2020) are followed for the computations.

Usage:

```
QCCS$JiGlobal.test(alpha = NULL)
```

Arguments:

`alpha` Significance level. By default a=0.05.

Returns: A list of class `htest` containing the results of the hypothesis test. In addition, the original data vectors and the probability vectors are also returned.

Examples:

```
Vectors<-list(c(18,0,3,0),c(27,19))
Prob<-list(c(0.85,0.1,0.03,0.02),c(0.8,0.2))
A <- QCCS$new(Vectors,Prob,
Source="Alba-Fernández et al. 2020")
A$JiGlobal.test()
```

References

Alba-Fernández MV, Ariza-López FJ, Rodríguez-Avi J, García-Balboa JL (2020). "Statistical methods for thematic-accuracy quality control based on an accurate reference sample." *Remote Sensing*, **12**(5), 816. Ariza-López FJ, Rodríguez-Avi J, Alba-Fernández MV, García-Balboa JL (2019). "Thematic accuracy quality control by means of a set of multinomials." *Applied Sciences*, **9**(20), 4240.

Examples

```
## ------------------------------------------------
## Method `QCCS$new()`
## ------------------------------------------------

Vectors<-list(c(47,4,0),c(44,5,3))
Prob<-list(c(0.95,0.04,0.01),c(0.88,0.1,0.02))
A<-QCCS$new(Vectors,Prob,
Source="Ariza-Lopez et al. 2019")


## ------------------------------------------------
## Method `QCCS$print()`
## ------------------------------------------------

Vectors<-list(c(18,0,3,0),c(27,19))
Prob<-list(c(0.85,0.1,0.03,0.02),c(0.8,0.2))
A<-QCCS$new(Vectors,Prob,
Source="Alba-Fernández et al. 2020")
A$print()


## ------------------------------------------------
## Method `QCCS$Exact.test()`
## ------------------------------------------------

Vectors<-list(c(47,4,0),c(40,5,3))
Prob<-list(c(0.95,0.04,0.01),c(0.88,0.1,0.02))
A<-QCCS$new(Vectors,Prob,
Source="Ariza-Lopez et al. 2019")
```

```
A$Exact.test()

## ------------------------------------------------
## Method `QCCS$Ji.test()`
## ------------------------------------------------

Vectors<-list(c(18,0,3,0),c(27,19))
Prob<-list(c(0.85,0.1,0.03,0.02),c(0.8,0.2))
A <- QCCS$new(Vectors,Prob,
Source="Alba-Fernández et al. 2020")
A$Ji.test()

## ------------------------------------------------
## Method `QCCS$JiGlobal.test()`
## ------------------------------------------------

Vectors<-list(c(18,0,3,0),c(27,19))
Prob<-list(c(0.85,0.1,0.03,0.02),c(0.8,0.2))
A <- QCCS$new(Vectors,Prob,
Source="Alba-Fernández et al. 2020")
A$JiGlobal.test()
```

Index

AvError, *2*
AvProdAcc, *2*
AvUserAcc, *2*

CombProdAcc_i, *2*
CombUserAcc_i, *2*
ConfMatrix, *2*, 3
ConfMatrix-package, 2

Error_i, *2*
Exact.test, *2*

f_i, *2*

g_i, *2*

h_i, *2*

Ji.test, *2*
JiGlobal.test, *2*

Kappa, *2*

Marghit, *2*

ProdAcc_i, *2*

QCCS, *2*, 51
Quasi_Independence, *2*

Relative_Agreement_Index, *2*

Tau, *2*

UserAcc_i, *2*