Package ‘PaolaR6Nuevo’

December 13, 2023

Type Package

Title Confusion Matrix

Version 0.1.0

Author Paola

Maintainer Paola Barba Ceballos <pbarba@ujaen.es>

Description This package provides useful functions for the analysis of confusion matrices in classification problems. Includes methods to calculate overall accuracy, user accuracy, and map creator accuracy.

Imports R6, Rdpack, ggplot2, gridExtra

RdMacros Rdpack

License GPL

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests knitr,
rmarkdown,
testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

R topics documented:

MatCon *Confusion matrix*

Description

Using the confusion matrix, various indices are calculated.

Value

Object of class MatCon.

Methods

Public methods:

- MatCon$new()
- MatCon$OverallAcc()
- MatCon$UserAcc()
- MatCon$UserAcc_i()
- MatCon$ProdAcc()
- MatCon$ProdAcc_i()
- MatCon$AvUserProdAcc_i()
- MatCon$Sucess()
- MatCon$Sucess_i()
- MatCon$AvHelldenAcc_i()
- MatCon$ShortAcc_i()
- MatCon$UserKappa_i()
- MatCon$ProdKappa_i()
- MatCon$ModKappa()
- MatCon$ModKappaUser_i()
- MatCon$ModKappaProd_i()
- MatCon$EntropUser_i()
- MatCon$EntropProd_i()
- MatCon$AvUserAcc()
- MatCon$AvProdAcc()
- MatCon$AvUserProdAcc()
- MatCon$AvHelldenAcc()
- MatCon$AvShortAcc()
- MatCon$CombUserAcc()
- MatCon$CombProdAcc()
- MatCon$CombUserProdAcc()
- MatCon$Kappa()
- MatCon$Entrop()
- MatCon$NormEntropUser()
- MatCon$NormEntropProd()
- MatCon$AvNormEntrop()
- MatCon$GeomAvNormEntrop()

- MatCon$AvMaxNormEntrop()
- MatCon$Tau()
- MatCon$UserProdAcc()
- MatCon$DetailedKappa()
- MatCon$DetailedCondKappa()
- MatCon$QES()
- MatCon$MTypify()
- MatCon$AllParameters()
- MatCon$MBootStrap()
- MatCon$MNormalize()
- MatCon$MPseudoZeroes()
- MatCon$DetailedWTau()
- MatCon$DetailedWKappa()
- MatCon$UserProdAcc_W()
- MatCon$StHell()
- MatCon$Kappa.test()
- MatCon$OverallAcc.test()
- MatCon$Tau.test()
- MatCon$TSCM.test()
- MatCon$plot.global()
- MatCon$plot.class()
- MatCon$clone()

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

Usage:

```
MatCon$new(values, ID = NULL, Date = NULL, Source = NULL)
```

Arguments:

`values` Confusion matrix

`ID` Identifier. By default, the date in YYYYMMDD format will be taken as the ID.

`Date` Date provided by the user. By default the date provided by the system will be taken.

`Source` Indicates where the matrix comes from (article, project, etc.). By default is NULL.

Returns: Object of class MatCon or an error if a matrix isn't entered.

Examples:

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

Method `OverallAcc()`**:** Public method to calculate the global index called Overall accuracy. The Overall accuracy for a particular classified image/map is then 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 samples taken. The method also offers the variance. The reference (Congalton and Green 2008) is followed for the computations.
The mathematical expression is:

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

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

Where:

1. $OverallAcc$: overall accuracy.
2. x_{ii}: diagonal element of the matrix.
3. x_{ij}: element of the matrix.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$OverallAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list of 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<-MatCon$new(A)
p$OverallAcc()
```

Method `UserAcc()`**:** Public method for deriving a class index called user's accuracy. The user's accuracy for the class i of thematic map 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. The method also offers the variance. The reference (Congalton and Green 2008) is followed for the computations.
The mathematical expression is:

$$UserAcc = \frac{x_{ii}}{\sum_{j=1}^{n} x_{ij}}$$

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

where:

1. $UserAcc$: user accuracy.
2. x_{ii}: diagonal element of the matrix.
3. x_{ij}: element of the matrix.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$UserAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with a vector of values for the user's accuracy rate for all classes, another vector with their variances and confidence intervals for each class.

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<-MatCon$new(A,Source="Congalton and Green 2008")
p$UserAcc()
```

Method `UserAcc_i()`**:** Public method where the user's accuracy index is defined for a specific class i. The user precision for class i of the thematic map is calculated by dividing the value on the diagonal of class i by the sum of all values in the row of class i. The method also offers variance. The reference (Congalton and Green 2008) is followed for the calculations.

$$UserAcc_i = \frac{x_{ii}}{\sum_{j=1}^{n} x_{ij}}$$

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

where:

1. $UserAcc_i$: user accuracy index for class i.
2. x_{ii}: diagonal element of the matrix.
3. x_{ij}: element of the matrix.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$UserAcc_i(i, a = NULL)
```

Arguments:

`i` User class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list of the user's accuracy index values 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$UserAcc_i(2)
```

Method `ProdAcc()`**:** Public method for deriving a class index called producer's accuracy. The producer's accuracy for the class i of thematic map 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. The method also offers the variance. The reference (Congalton and Green 2008) if followed for the computations.

$$ProdAcc = \frac{x_{jj}}{\sum_{j=1}^{n} x_{ij}}$$

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

where:

1. $ProdAcc$: producer accuracy.
2. x_{jj}: diagonal element of the matrix.
3. x_{ij}: element of the matrix.

4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$ProdAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with a vector of values for the producer's accuracy index of all classes, another vector with their variances and confidence intervals for each class.

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<-MatCon$new(A,Source="Congalton and Green 2008")
p$ProdAcc()
```

Method `ProdAcc_i()`**:** Public method where the producer's accuracy index is defined for a specific class i. The user precision for class i of the thematic map is calculated by dividing the value on the diagonal of class i by the sum of all values in the column of class i. The method also offers variance. The reference (Congalton and Green 2008) is followed for the calculations.

$$ProdAcc_i = \frac{x_{jj}}{\sum_{j=1}^{n} x_{ij}}$$

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

where:

1. $ProdAcc_i$: producer accuracy index for class i.
2. x_{jj}: diagonal element of the matrix.
3. x_{ij}: element of the matrix.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$ProdAcc_i(i, a = NULL)
```

Arguments:

`i` Producer class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list of the producer's accuracy index values 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$ProdAcc_i(1)
```

Method `AvUserProdAcc_i()`**:** Public method that provides the average of the accuracy rates of the user and producer of a specific class. The method also offers variance. The reference (Liu et al. 2007) is followed for the calculations.

The mathematical expression is:

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

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

where:

1. $AvUserProdAcc_i$: average of user's and producer's accuracy.
2. $UserAcc_i$: user accuracy index for class i.
3. $ProdAcc_i$: producer accuracy index for class i.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvUserProdAcc_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with average of user's and producer's accuracy, its variance for class i 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc_i(2)
```

Method `Sucess()`**:** Public method that provides the Classification Success Index (CSI) applies to all class and gives an overall estimation of classification effectiveness. The references (Koukoulas and Blackburn 2001; Turk 2002) is followed for the calculations.
The mathematical expression is:

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

$$VarSucess = \frac{Sucess \cdot (1 - Sucess)}{N}$$

where:

1. $Sucess$: classification succes index.
2. $AvUserAcc$: average accuracy from user's perspective.
3. $AvProdAcc$: average accuracy from producer's perspective.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$Sucess(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with the classification success index, 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<-MatCon$new(A,Source="Labatut and Cherifi 2011")
p$Sucess()
```

Method `Sucess_i()`**:** Public method that provides the Individual Classification Success Index (ICSI) applies to the classification effectiveness for one particular class of interest. The references (Koukoulas and Blackburn 2001; Turk 2002) is followed for the calculations.

The mathematical expression is:

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

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

where:

1. $Sucess_i$: individual classification success index.
2. $UserAcc_i$: user accuracy index for class i.
3. $ProdAcc_i$: producer accuracy index for class i.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$Sucess_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with the individual classification success index, 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<-MatCon$new(A,Source="Labatut and Cherifi 2011")
p$Sucess_i(2)
```

Method `AvHelldenAcc_i()`**:** Public method that provides the Hellden' average accuracy, denotes for the probability that a randomly chosen point of a specific class on the map has a correspondence of the same class in the same position in the field and that a randomly chosen point in the field of the same class has a correspondence of the same class in the same position on the map.The method also offers variance. The references (Helldén 1980; Rosenfield and Fitzpatrick-Lins 1986) is followed for the calculations.

$$AvHelldenAcc_i = \frac{2}{\frac{1}{UserAcc_i} + \frac{1}{ProdAcc_i}}$$

$$\sigma^2_{AvHelldenAcc_i} = \frac{AvHelldenAcc_i \cdot (1 - AvHelldenAcc_i)}{N}$$

where:

1. $AvHelldenAcc_i$: Hellden's mean accuracy.
2. $UserAcc_i$: user accuracy index for class i.
3. $ProdAcc_i$: producer accuracy index for class i.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvHelldenAcc_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with 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<-MatCon$new(A,Source="Rosenfield and Fitzpatrick 1986")
p$AvHelldenAcc_i(2)
```

Method `ShortAcc_i()`**:** Public method that provides Short's mapping accuracy for each class is stated as the number of correctly classified pixels (equal to the total in the correctly classified area) in terms of all pixels affected by its classification (equal to this total in the displayed area as well as the pixels involved in errors of commission and omission). The method also offers variance. The references (Rosenfield and Fitzpatrick-Lins 1986; Short 1982) is followed for the calculations.

$$ShortAcc_i = \frac{x_{ii}}{\sum_{j=1}^{n} x_{+j} + \sum_{i=1}^{n} x_{i+} - x_{ii}}$$

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

where:

1. $ShortAcc_i$: Short's mapping accuracy
2. x_{ii}: diagonal element of the matrix.
3. x_{j+}: sum of all elements in rows j.
4. x_{+j}: sum of all elements in column j.
5. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$ShortAcc_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with 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<-MatCon$new(A,Source="Rosenfield and Fitzpatrick-Lins 1986")
p$ShortAcc_i(2)
```

Method `UserKappa_i()`**:** Public method that evaluates the kappa coefficient from the user's perspective, for a specific class i. The method also offers variance. The reference (Rosenfield and Fitzpatrick-Lins 1986) is followed for the calculations.

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

$$\sigma^2_{UserKappa_i} = \frac{UserKappa_i \cdot (1 - UserKappa_i)}{N}$$

where:

1. $UserKappa_i$: coefficient kappa (user's).
2. $UserAcc_i$: user accuracy index for class i.
3. x_{ii}: diagonal element of the matrix.
4. x_{j+}: sum of all elements in rows j.
5. x_{+j}: sum of all elements in column j.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$UserKappa_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with coefficient kappa (user'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<-MatCon$new(A,Source="Næsset 1996")
p$UserKappa_i(2)
```

Method `ProdKappa_i()`**:** Public method that evaluates the kappa coefficient from the producer's perspective, for a specific class i. The method also offers variance. The reference (Rosenfield and Fitzpatrick-Lins 1986) is followed for the calculations.

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

$$\sigma^2_{ProdKappa_i} = \frac{ProdKappa_i \cdot (1 - ProdKappa_i)}{N}$$

where:

1. $ProdKappa_i$: coefficient kappa (producer's).
2. $ProdAcc_i$: producer accuracy index for class i.
3. x_{ii}: diagonal element of the matrix.
4. x_{j+}: sum of all elements in rows j.
5. x_{+j}: sum of all elements in column j.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$ProdKappa_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with 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<-MatCon$new(A,Source="Næsset 1996")
p$ProdKappa_i(2)
```

Method `ModKappa()`**:** Public method that provides the overall modified kappa coefficient. The method also offers variance. The references (Stehman 1997; Foody 1992) is followed for the calculations.

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

$$\sigma^2_{ModKappa} = \frac{ModKappa \cdot (1 - ModKappa)}{N}$$

where:

1. $ModKappa$: modified coefficient kappa.
2. $OverallAcc$: overall accuracy.
3. M: number of elements of the matrix.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$ModKappa(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with 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<-MatCon$new(A,Source="Foody 1992")
p$ModKappa()
```

Method `ModKappaUser_i()`**:** Public method, derived from the general modified kappa coefficient, which provides the modified coefficient kappa for the user. The method also offers variance. The references (Stehman 1997; Foody 1992) is followed for the calculations.

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

$$\sigma^2_{ModKappaUser_i} = \frac{ModKappaUser_i \cdot (1 - ModKappaUser_i)}{N}$$

where:

1. $ModKappaUser_i$: modified coefficient kappa (user's).
2. $UserAcc_i$: user accuracy index for class i.
3. M: number of elements of the matrix.

4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$ModKappaUser_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with modified coefficient kappa (user's), 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<-MatCon$new(A,Source="Liu et al. 2007")
p$ModKappaUser_i(2)
```

Method `ModKappaProd_i()`**:** Public method, derived from the general modified kappa coefficient, which provides the modified coefficient kappa for the producer. The method also offers variance. The references (Stehman 1997; Foody 1992) is followed for the calculations.

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

$$\sigma^2_{ModKappaProd_i} = \frac{ModKappaProd_i \cdot (1 - ModKappaProd_i)}{N}$$

where:

1. $ModKappaUser_i$: modified coefficient kappa (producer's).
2. $ProdAcc_i$: producer accuracy index for class i.
3. M: number of elements of the matrix.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$ModKappaProd_i(i, a = NULL)
```

Arguments:

`i` Class to evaluate.

`a` Significance level. By default 0.05.

Returns: A list with modified coefficient kappa (producer's), 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<-MatCon$new(A,Source="Liu et al. 2007")
p$ModKappaProd_i(2)
```

Method `EntropUser_i()`**:** Public method that calculates relative change of entropy given a category on map. That is, the degree of uncertainty of the category. The method also offers variance. The reference (Finn 1993) is followed for the calculations.

$$Entrop_i(A) = -\sum_{j=1}^{n}((\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$Entrop_i(A|b_i) = -\sum_{j=1}^{n}((\frac{x_{ij}}{\sum_{j=1}^{n} x_{+j}}) \cdot \log(\frac{x_{ij}}{\sum_{j=1}^{n} x_{+j}}))$$

$$EntropUser_i = \frac{Entrop_i(A) - Entrop_i(A|b_i)}{Entrop_i(A)}$$

$$\sigma^2_{EntropUser_i} = \frac{EntropUser_i \cdot (1 - EntropUser_i)}{N}$$

where:

1. $EntropUser_i$: relative change of entropy given a category on map.
2. $Entrop_i(A)$: Entropy of the map with respect to the category of the map.
3. x_{j+}: sum of all elements in rows j.
4. x_{+j}: sum of all elements in column j.
5. $Entrop_i(A|b_i)$: Entropy of map A knowing that the location corresponding to map B is in class b_i.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$EntropUser_i(i, v = NULL, a = NULL)
```

Arguments:

`i` Class to evaluate (row).

`v` Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

`a` Significance level. By default 0.05.

Returns: A list with the relative change of entropy given a category on map, its variance, its confidence interval, map entropy, and entropy of map A knowing that the location corresponding to map B is in class b_i.

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<-MatCon$new(A,Source="Liu et al. 2007")
p$EntropUser_i(1)
```

Method `EntropProd_i()`**:** Public method that calculates relative change of entropy given a category on ground truthing. That is, the degree of uncertainty of the category. The method also offers variance. The reference (Stehman 1997) is followed for the calculations.

$$Entrop_i(B) = -\sum_{i=1}^{n}((\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$Entrop_i(B|a_j) = -\sum_{j=1}^{n}((\frac{x_{ij}}{\sum_{i=1}^{n} x_{i+}}) \cdot \log(\frac{x_{ij}}{\sum_{i=1}^{n} x_{i+}}))$$

$$EntropProd_i = \frac{EntropMap(B) - EntropMap(B|a_j)}{EntropMap(B)}$$

$$\sigma^2_{EntropProd_i} = \frac{EntropProd_i \cdot (1 - EntropProd_i)}{N}$$

where:

1. $EntropProd_i$: relative change of entropy given a category on ground truthing.
2. $Entrop_i(B)$: Entropy of the map with respect to the category on ground truthing.
3. x_{j+}: sum of all elements in rows j.
4. x_{+j}: sum of all elements in column j.
5. $Entrop_i(B|a_j)$: Entropy of map B knowing that the location corresponding to map A is in class a_j.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$EntropProd_i(i, v = NULL, a = NULL)
```

Arguments:

`i` Class to evaluate

`v` Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

`a` Significance level. By default 0.05.

Returns: A list of the relative change of entropy given a category on ground truthing, its variance,its confidence interval, map entropy, and entropy of map B knowing that the location corresponding to map A is in class a_j.

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<-MatCon$new(A,Source="Liu et al. 2007")
p$EntropProd_i(2)
```

Method `AvUserAcc()`**:** Public method that provides the user's average accuracy, which is an average of the accuracy of individual categories, in this case the categories will be taken from the user's perspective. The method also offers variance. The reference (Tung and LeDrew 1988) is followed for the calculations.

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

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

where:

1. $AvUserAcc$: average accuracy from user's perspective.
2. x_{j+}: sum of all elements in rows j.
3. x_{ii}: diagonal element of the matrix.
4. M: number of elements of the matrix.
5. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvUserAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with the average accuracy from user's perspective, its variance and its confidence interval.

Examples:

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

Method `AvProdAcc()`**:** Public method that provides the producer's average accuracy, which is an average of the accuracy of individual categories, in this case the categories will be taken from the producer's perspective. The method also offers variance. The reference (Tung and LeDrew 1988) is followed for the calculations.

$$AvProdAcc = \frac{1}{\sqrt{N}} \sum_{i=1}^{n} \frac{x_{ii}}{\sum_{j=1}^{n} x_{+j}}$$

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

where:

1. $AvProdAcc$: average accuracy from producer's perspective.
2. x_{+j}: sum of all elements in column j.
3. x_{ii}: diagonal element of the matrix.
4. M: number of elements of the matrix.
5. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvProdAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with the average accuracy from producer's perspective, its variance and its confidence interval.

Examples:

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

Method `AvUserProdAcc()`**:** Public method that offers the average of the average precision from the perspective of the user and the producer. The method also offers variance. The reference (Liu et al. 2007) is followed for the calculations.

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

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

where:

1. $AvUserProdAcc$: average of average of user's and producer's perspective.
2. $AvUserAcc$: average accuracy from user's perspective.
3. $AvProdAcc$: average accuracy from producer's perspective.

4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvUserProdAcc(a = NULL)
```

Arguments:

a Significance level. By default 0.05.

Returns: A list of the average mean precision values from the user and producer 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc()
```

Method `AvHelldenAcc()`**:** Public method that provides the average value of the Hellden mean precision index. The method also offers variance. The reference (Liu et al. 2007) is followed for the calculations.

$$AvHelldenAcc = \frac{1}{\sqrt{M}} \sum_{i=1}^{n} \frac{2x_{ii}}{x_{+i} + x_{i+}}$$

$$\sigma^2_{AvHelldenAcc} = \frac{AvHelldenAcc \cdot (1 - AvHelldenAcc)}{N}$$

where:

1. $AvHelldenAcc$: average of Hellden's mean accuracy index.
2. x_{+i}: sum of all elements in column i.
3. x_{i+}: sum of all elements in row i.
4. x_{ii}: diagonal element of the matrix.
5. M: number of elements of the matrix.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvHelldenAcc(a = NULL)
```

Arguments:

a Significance level. By default 0.05.

Returns: A list with 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvHelldenAcc()
```

Method `AvShortAcc()`**:** Public method that provides the average of Short's mapping accuracy index. The method also offers variance. The reference (Liu et al. 2007) is followed for the calculations.

$$AvShortAcc = \frac{1}{\sqrt{M}} \frac{\frac{\sum_{i=1}^{n} x_{ii}}{\sum_{i,j=1}^{n} x_{ij}}}{\sum_{j=1}^{n} x_{+j} + \sum_{i=1}^{n} x_{i+} - x_{ii}}$$

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

where:

1. x_{+i}: sum of all elements in column i.
2. x_{i+}: sum of all elements in row i.
3. x_{ii}: diagonal element of the matrix.
4. M: number of elements of the matrix.
5. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvShortAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvShortAcc()
```

Method `CombUserAcc()`**:** Public method that provides the combined user accuracy that is the average of the overall accuracy and the average user accuracy. The method also offers variance. The reference (Tung and LeDrew 1988) is followed for the calculations.

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

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

where:

1. $CombUserAcc$: combined accuracy from user's perspective.
2. $OverallAcc$: overall accuracy.
3. $AvUserAcc$: average accuracy from user's perspective.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$CombUserAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list of 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<-MatCon$new(A,Source="Tung and LeDrew 1988")
p$CombUserAcc()
```

Method `CombProdAcc()`**:** Public method that provides the combined producer accuracy that is the average of the overall accuracy and the average producer accuracy. The method also offers variance. The reference (Tung and LeDrew 1988) is followed for the calculations.

$$CombProdAcc = \frac{OverallAcc + AvProdAcc}{2}$$
$$\sigma^2_{CombProdAcc} = \frac{CombProdAcc \cdot (1 - CombProdAcc)}{N}$$

where:

1. $CombProdAcc$: combined accuracy from producer's perspective.
2. $OverallAcc$: overall accuracy.
3. $AvProdAcc$: average accuracy from producer's perspective.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$CombProdAcc(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list of 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<-MatCon$new(A,Source="Tung and LeDrew 1988")
p$CombProdAcc()
```

Method `CombUserProdAcc()`**:** Public method that provides the combined accuracy which is the average of the overall accuracy and the Hellden average accuracy, which refers to the average user and producer accuracies. The method also offers variation. The reference (Liu et al. 2007) is followed for the calculations.

$$CombUserProdAcc = \frac{OverallAcc + AvHelldenAcc}{2}$$
$$\sigma^2_{CombUserProdAcc} = \frac{CombUserProdAcc \cdot (1 - CombUserProdAcc)}{N}$$

where:

1. $CombUserProdAcc$: combined accuracy from both user's and producer's perspectives.
2. $OverallAcc$: overall accuracy.
3. $AvHelldenAcc$: average of Hellden's mean accuracy index.
4. N: number of cases involved in the calculation of the index.

Usage:

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

Arguments:

`a` Significance level. By default 0.05.

Returns: A list of 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$CombUserProdAcc()
```

Method `Kappa()`**:** Public method that provides kappa coefficient, which measures the relationship between agreement beyond chance and expected disagreement. The method also offers variation. The reference (Cohen 1960) is followed for the calculations.

$$ExpAcc = \sum_{i=1}^{n} (\frac{x_{+i}}{\sum_{j=1}^{n} x_{ij}} \cdot \frac{x_{i+}}{\sum_{j=1}^{n} x_{ij}})$$
$$Kappa = \frac{OverallAcc - ExpAcc}{1 - ExpAcc}$$
$$\sigma^2_{Kappa} = \frac{OverallAcc - ExpAcc}{(1 - ExpAcc) \cdot N}$$

where:

1. $Kappa$: Kappa coefficient.
2. $OverallAcc$: overall accuracy.
3. $ExpAcc$: expected accuracy of agreement if agreement were purely random.
4. x_{+i}: sum of all elements in column i.
5. x_{i+}: sum of all elements in row i.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$Kappa(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$Kappa()
```

Method `Entrop()`**:** Public method for calculating map entropy. Which refers to the degree of uncertainty that the map presents. The method also offers variation. The reference (Finn 1993) is followed for the calculations.

$$Entrop = \sum_{i,j=1}^{n} (\frac{x_{ij}}{\sum_{i,j=1}^{n} x_{ij}} \cdot \log(\frac{x_{ij}}{\frac{\sum_{i=1}^{n} x_{i+} \cdot \sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}}))$$
$$\sigma^2_{Entrop} = \frac{Entrop \cdot (1 - Entrop)}{N}$$

where:

1. $Entrop$: map entropy.
2. x_{+i}: sum of all elements in column i.
3. x_{i+}: sum of all elements in row i.
4. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$Entrop(v = NULL, a = NULL)
```

Arguments:

`v` Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

`a` Significance level. By default 0.05.

Returns: A list with map entropy, 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<-MatCon$new(A,Source="Liu et al. 2007")
p$Entrop()
```

Method `NormEntropUser()`**:** Public method that calculates normalized entropy using the map. The method also offers variation. The reference (Finn 1993) is followed for the calculations.

$$Entrop_i(B) = -\sum_{i=1}^{n}((\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$NormEntropUser = \frac{Entrop}{Entrop_i(B)}$$

$$\sigma^2_{NormEntropUser} = \frac{NormEntropUser \cdot (1 - NormEntropUser)}{N}$$

where:

1. $NormEntropUser$: normalized entropy using map.
2. $Entrop_i(B)$: entropy of the map with respect to the category on ground truthing.
3. $Entrop$: map entropy.
4. x_{+i}: sum of all elements in column i.
5. x_{i+}: sum of all elements in row i.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$NormEntropUser(v = NULL, a = NULL)
```

Arguments:

`v` Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

`a` Significance level. By default 0.05.

Returns: A list with normalized entropy using map, 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<-MatCon$new(A,Source="Liu et al. 2007")
p$NormEntropUser()
```

Method `NormEntropProd()`: Public method that calculates normalized entropy using on ground truthing. The method also offers variation. The reference (Finn 1993) is followed for the calculations.

$$Entrop_i(A) = -\sum_{j=1}^{n}((\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$NormEntropProd = \frac{Entrop}{Entrop_i(A)}$$

$$\sigma^2_{NormEntropProd} = \frac{NormEntropProd \cdot (1 - NormEntropProd)}{N}$$

where:

1. $NormEntropProd$: normalized mutual information using the entropy on ground truthing.
2. $Entrop_i(A)$: Entropy of the map with respect to the category of the map.
3. $Entrop$: map entropy.
4. x_{+i}: sum of all elements in column i.
5. x_{i+}: sum of all elements in row i.
6. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$NormEntropProd(v = NULL, a = NULL)
```

Arguments:

`v` Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

`a` Significance level. By default 0.05.

Returns: A list with normalized entropy using on ground truthing, 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<-MatCon$new(A,Source="Liu et al. 2007")
p$NormEntropProd()
```

Method `AvNormEntrop()`: Public method that calculates normalized entropy using the arithmetic mean of the entropies on the map and on ground truthing. The method also offers variation. The reference (Strehl and Ghosh 2002) is followed for the calculations.

$$Entrop_i(A) = -\sum_{j=1}^{n}((\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$Entrop_i(B) = -\sum_{i=1}^{n}((\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$AvNormEntrop = \frac{2Entrop}{Entrop_i(A) + Entrop_i(B)}$$

$$\sigma^2_{AvNormEntrop} = \frac{AvNormEntrop \cdot (1 - AvNormEntrop)}{N}$$

where:

1. $AvNormEntrop$: normalized entropy using the arithmetic mean of the entropies on the map and on ground truthing.
2. $Entrop_i(B)$: entropy of the map with respect to the category on ground truthing.
3. $Entrop_i(A)$: Entropy of the map with respect to the category of the map.
4. $Entrop$: map entropy.
5. x_{+i}: sum of all elements in column i.
6. x_{i+}: sum of all elements in row i.
7. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvNormEntrop(v = NULL, a = NULL)
```

Arguments:

v Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

a Significance level. By default 0.05.

Returns: normalized entropy using the arithmetic mean of the entropies on the map and on ground truthing, 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvNormEntrop()
```

Method `GeomAvNormEntrop()`**:** Public method that calculates normalized entropy using the geometric mean of the entropies on the map and on ground truthing. The method also offers variation. The reference (Ghosh et al. 2002) is followed for the calculations.

$$Entrop_i(A) = -\sum_{j=1}^{n}((\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$Entrop_i(B) = -\sum_{i=1}^{n}((\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$GeomAvNormEntrop = \frac{Entrop}{\sqrt{Entrop_i(A) \cdot Entrop_i(B)}}$$

$$\sigma^2_{GeomAvNormEntrop} = \frac{GeomAvNormEntrop \cdot (1 - GeomAvNormEntrop)}{N}$$

where:

1. $GeomAvNormEntrop$: normalized entropy using the geometric mean of the entropies on map and on ground truthing.
2. $Entrop_i(B)$: entropy of the map with respect to the category on ground truthing.
3. $Entrop_i(A)$: Entropy of the map with respect to the category of the map.
4. $Entrop$: map entropy.
5. x_{+i}: sum of all elements in column i.
6. x_{i+}: sum of all elements in row i.
7. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$GeomAvNormEntrop(v = NULL, a = NULL)
```

Arguments:

v Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

a Significance level. By default 0.05.

Returns: A list with normalized entropy using the geometric mean of the entropies on map and on ground truthing, 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$GeomAvNormEntrop()
```

Method `AvMaxNormEntrop()`**:** Public mathod that provides normalized entropy using the arithmetic mean of the maximum entropies on map and on ground truthing.The method also offers variation. The reference (Strehl 2002) is followed for the calculations.

$$Entrop_i(A) = -\sum_{j=1}^{n}((\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{i=1}^{n} x_{i+}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$Entrop_i(B) = -\sum_{i=1}^{n}((\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}) \cdot \log(\frac{\sum_{j=1}^{n} x_{+j}}{\sum_{i,j=1}^{n} x_{ij}}))$$

$$AvMaxNormEntrop = \frac{2Entrop}{max(Entrop_i(A)) + max(Entrop_i(B))} = \frac{Entrop}{\log\sqrt{M}}$$

where:

1. $AvMaxNormEntrop$: normalized entropy using the arithmetic mean of the maximum entropies on map and on ground truthing.
2. $Entrop_i(B)$: entropy of the map with respect to the category on ground truthing.
3. $Entrop_i(A)$: Entropy of the map with respect to the category of the map.
4. $Entrop$: map entropy.
5. x_{+i}: sum of all elements in column i.
6. x_{i+}: sum of all elements in row i.
7. M: number of elements of the matrix.
8. N: number of cases involved in the calculation of the index.

Usage:

```
MatCon$AvMaxNormEntrop(v = NULL, a = NULL)
```

Arguments:

v Base of the logarithm. By default v=10. This value is used for the entropy units, v=10(Hartleys), v=2(bits), v=e(nats).

a Significance level. By default 0.05.

Returns: A list with normalized entropy using the arithmetic mean of the maximum entropies on map and on ground truthing, 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvMaxNormEntrop()
```

Method `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. The reference (Ariza-Lopez et al. 2013) is followed for the computations.
The mathematical expression is:

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

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

$$\sigma^2_{Tau} = \frac{OverallAcc \cdot (1 - OverallAcc)}{N \cdot (1 - CoefAccPr)^2}$$

Where:

1. $OverallAcc$: overall accuracy.
2. $PrAgCoef$: a priori random agreement coefficient.
3. M: number of classes.
4. N: number of elements of the matrix, cardinal of the matrix.

Usage:

```
MatCon$Tau(a = NULL)
```

Arguments:

`a` Significance level. By default 0.05.

Returns: A list with 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<-MatCon$new(A,Source="Muñoz 2016")
p$Tau()
```

Method `UserProdAcc()`**:** Public method that calculates the pressures of the user and the producer jointly. The method also offers the standard desviations. The reference (Congalton and Green 2008) is followed for the computations.

Usage:

```
MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$UserProdAcc()
```

Method `DetailedKappa()`**:** Public method that calculates the general Kappa agreement index, its standard deviation and the test statistic to test its significance. The reference (Congalton and Green 2008) is followed for the computations.

Usage:

```
MatCon$DetailedKappa()
```

Returns: A list of 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$DetailedKappa()
```

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

Usage:

```
MatCon$DetailedCondKappa()
```

Returns: A list with conditional Kappa index of the user and the producer, and its corresponding standard deviation.

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<-MatCon$new(A,Source="Congalton and Green 2008")
p$DetailedCondKappa ()
```

Method `QES()`**:** Public method that calculates the values of quantity, change and shift. The reference (Pontius Jr and Santacruz 2014) is followed for the computations.

Usage:

```
MatCon$QES(TI = NULL, SF = 1)
```

Arguments:

`TI` Time interval (default value = 1)

`SF` Scale factor for results (default value = 1)

Returns: A list of general values for the interval t of difference, quantity, shift, and shift.In addition to the differences for categories, number of components, change of categories and turn of the components.

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<-MatCon$new(A,Source="Congalton and Green 2008")
p$QES(TI=1, SF=6)
```

Method `MTypify()`**:** Public method that types the values of each cell. The total sum of the original matrix is used for typing. The resulting values can be presented as real (parameter RaR=1) or as a percentage (parameter RaR !=1)

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

where:

1. MTyipify: typified matrix.
2. x_{ij}: matrix element.

Usage:

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

Arguments:

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

Returns: A list with original matrix and typified 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<-MatCon$new(A, Source="Congalton and Green 2008")
p$MTypify(RaR=5)
```

Method `AllParameters()`**:** Public method in which multiple parameters are calculated for the given confusion matrix. The references (Congalton and Green 2008; Cohen 1960; Muñoz 2016) is followed for the computations.

Usage:

```
MatCon$AllParameters()
```

Returns: A list containing confusion matrix, dimension, total sum of cell values, overall precision, overall variance precision, global precision kappa index, global kappa simplified variance, producer precision by class, user precision by class, 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AllParameters()
```

Method `MBootStrap()`**:** Public method that provides N resamples of the confusion matrix from a MatCon object. The reference (Ariza et al. 2011) is followed for the computations.

Usage:

```
MatCon$MBootStrap(n, pr = NULL)
```

Arguments:

`n` Number of resamples.

`pr` Probability for resampling. By default, the probability of success for each cell will be taken.

Returns: A list formed by the original confusion matrix and simulated matrices, from the confusion matrix. The multinomial distribution is applied.

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<-MatCon$new(A, Source="Congalton and Green 2008")
p$MBootStrap(2)
```

Method `MNormalize()`**:** Public method that carries out an iterative process is carried out where each element is divided by the total of the sum of its row, thus obtaining new values. In the next iteration, all the elements are added by columns and each element is divided by the total of its column and they obtain new values, and so on. The references (Fienberg 1970; Muñoz 2016) is followed for the computations.

Usage:

```
MatCon$MNormalize(n = NULL)
```

Arguments:

`n` Number of iteration. By default n=100.

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<-MatCon$new(A,Source="Muñoz 2016")
p$MNormalize()$values
```

Method `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.

Usage:

```
MatCon$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<-MatCon$new(A,Source="Muñoz 2016")
p$MPseudoZeroes()
```

Method `DetailedWTau()`**:** Public method that calculates the general Tau concordance index and its standard deviation.

Usage:

```
MatCon$DetailedWTau(WV)
```

Arguments:

`WV` Weights vector (as matrix)

Returns: A list with the weight matrix, the Tau index, 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$DetailedWTau(WV)
```

Method `DetailedWKappa()`**:** Public method that calculates the general Kappa agreement index (weighted) and its standard deviation. The reference (Congalton and Green 2008) is followed for the computations.

Usage:

```
MatCon$DetailedWKappa(WM)
```

Arguments:

`WM` Weight matrix

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

Examples:

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

Method `UserProdAcc_W()`**:** Public method that calculates the weighted accuracies and standard deviations of the user and the producer. The reference (Congalton and Green 2008) is followed for the computations.

Usage:

```
MatCon$UserProdAcc_W(WM)
```

Arguments:

`WM` Weight matrix

Returns: A list with weight matrix, Matrix formed with its original elements and their corresponding weights, general accuracy of the weight matrix obtained, accuracy of the producer and user 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<-MatCon$new(A,Source="Congalton and Green 2008")
WM<- t(matrix(c(1,0,0.67,1,0,1,0,0,1,0,1,1,0.91,0,0.61,1),
nrow = 4, ncol=4))
p$UserProdAcc_W(WM)
```

Method `StHell()`**:** Public method that provides the Hellinger distance between two elements of the MatCon class. The reference (García-Balboa et al. 2018) is followed for the computations. The mathematical expression is:

$$HD = \frac{4nm}{n+m} \sum_{i=1}^{M} (\sqrt{p_i} - \sqrt{q_i})^2$$

Where:

1. HD: Hellinger Distance
2. n: number of elements in the matrix A.
3. p_i: element i of the probability vector of matrix A.
4. q_i: element i of the probability vector of matrix B.

Usage:

```
MatCon$StHell(f, p = NULL, q = NULL)
```

Arguments:

`f` f Element of the MatCon.

`p` matrix probability vector. By default, the probability of success for each cell is taken.

`q` matrix probability vector. By default, the probability of success for each cell is taken.

Returns: The statistic value of the statistical test based on 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<-MatCon$new(A,Source="Congalton and Green 2008")
B<-matrix(c(45,6,0,4,4,91,8,7,12,5,55,3,24,8,9,55),
nrow=4,ncol=4)
f<-MatCon$new(B,Source="Congalton and Green 2008")
p$StHell(f)
```

Method `Kappa.test()`**:** Public method that tests whether two independent confusion matrices of the MatCon class are significantly different using their kappa index. The reference (Congalton and Green 2008) is followed for the computations. The mathematical expression to calculate its statistic is:

$$Z = \frac{|k1 - k2|}{\sqrt(var(K1) + var(K2))}$$

Where:

1. $k1$: kappa index of matrix A
2. $k2$: kappa index of matrix B
3. $var(k1)$: variance of k1.
4. $var(k2)$: variance of k2.

Usage:

```
MatCon$Kappa.test(f, alpha = NULL)
```

Arguments:

`f` Element of the MatCon class.

`alpha` significance level. By default alpha=0.05.

Returns: A list with the value of the statistic between kappa values and its z score for a given alpha significance level.

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

Method `OverallAcc.test()`**:** Public method that tests whether two independent confusion matrices of the MatCon class are significantly different using their overall accuracy index. The reference (Ariza-Lopez et al. 2013; Ma and Redmond 1995) is followed for the computations. The mathematical expression to calculate its statistic is:

$$Z = \frac{|k1 - k2|}{\sqrt{var(k1) + var(k2)}}$$

Where:

1. $k1$: overall index of matrix A
2. $k2$: overall index of matrix B
3. $var(K1)$: variance of k1.
4. $var(K2)$: variance of k2.

Usage:

```
MatCon$OverallAcc.test(f, alpha = NULL)
```

Arguments:

`f` Element of the MatCon class.

`alpha` significance level. By default alpha=0.05.

Returns: A list of the statistic's value between the overall accuracies and its z-score for a given alpha significance level.

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

Method `Tau.test()`**:** Public method that tests whether two independent confusion matrices of the MatCon class are significantly different using their Tau index. The reference (Ariza-Lopez et al. 2013; Ma and Redmond 1995) is followed for the computations. The mathematical expression to calculate its statistic is:

$$Z = \frac{|k1 - k2|}{\sqrt{var(k1) + var(k2)}}$$

Where:

1. $k1$: Tau index of matrix A
2. $k2$: Tau index of matrix B
3. $var(k1)$: variance of k1.
4. $var(k2)$: variance of k2.

Usage:

```
MatCon$Tau.test(f, alpha = NULL)
```

Arguments:

`f` Element of the MatCon class.

`alpha` significance level. By default alpha=0.05.

Returns: A list of the statistic's value between the Tau index and its z-score for a given alpha significance level.

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

Method `TSCM.test()`**:** Public method that performs a homogeneity test between two matrices of the MatCon class based on the Hellinger distance. The test considers the individual cell values in the matrices. The reference (García-Balboa et al. 2018) is followed for the computations.

Usage:

```
MatCon$TSCM.test(f, n1 = NULL, alpha = NULL)
```

Arguments:

`f` Element of the MatCon class.

`n1` Number of bootstraps that you want to generate. By default n=10000.

`alpha` significance level. By default alpha=0.05.

Returns: p value, alpha and decision to make.

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

Method `plot.global()`**:** Public method that provides the graph of the indices of the functions OverallAcc, Kappa, Tau, GroundTruth, AvHelldenAcc, AvShortAcc with their corresponding standard desviation.

Usage:

```
MatCon$plot.global()
```

Returns: the graph of the indices of the functions OverallAcc, Kappa, Tau, GroundTruth, AvHelldenAcc, AvShortAcc 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<-MatCon$new(A,Source="Congalton and Green 2008")
p$plot.global()
```

Method `plot.class()`**:** Public method that provides the graph of the accuracy index of users and producers with their corresponding standard desviation.

Usage:

```
MatCon$plot.class()
```

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<-MatCon$new(A,Source="Congalton and Green 2008")
p$plot.class()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
MatCon$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Note

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.

References

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

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

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.

Turk G (2002). "Map evaluation and" chance correction"." *Photogrammetric Engineering and Remote Sensing*, **68**(2), 123–+.

Helldén 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*.

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~. . . .

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.

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.

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

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

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.

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

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 Ja\'e. Servicio de publicaciones. ISBN 9788484398134.

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.

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

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

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

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.

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

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.

Examples

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


## ------------------------------------------------
## Method `MatCon$new`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A)
p$OverallAcc()


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$UserAcc()


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$UserAcc_i(2)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$ProdAcc()


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$ProdAcc_i(1)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc_i(2)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Labatut and Cherifi 2011")
p$Sucess()


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Labatut and Cherifi 2011")
p$Sucess_i(2)


## ------------------------------------------------
## Method `MatCon$AvHelldenAcc_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<-MatCon$new(A,Source="Rosenfield and Fitzpatrick 1986")
p$AvHelldenAcc_i(2)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Rosenfield and Fitzpatrick-Lins 1986")
p$ShortAcc_i(2)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Næsset 1996")
p$UserKappa_i(2)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Næsset 1996")
p$ProdKappa_i(2)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Foody 1992")
p$ModKappa()
```

```
## ------------------------------------------------
## Method `MatCon$ModKappaUser_i`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Liu et al. 2007")
p$ModKappaProd_i(2)


## ------------------------------------------------
## Method `MatCon$EntropUser_i`
## ------------------------------------------------

A<-matrix(c(0,12,0,0,12,0,0,0,0,0,12,0,0,12,0),
nrow=4,ncol=4)
p<-MatCon$new(A,Source="Liu et al. 2007")
p$EntropUser_i(1)


## ------------------------------------------------
## Method `MatCon$EntropProd_i`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$AvUserAcc`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$AvProdAcc`
## ------------------------------------------------

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

```
## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvUserProdAcc()


## ------------------------------------------------
## Method `MatCon$AvHelldenAcc`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AvShortAcc()


## ------------------------------------------------
## Method `MatCon$CombUserAcc`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$CombProdAcc`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$CombUserProdAcc()
```

```
## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$Kappa()


## ------------------------------------------------
## Method `MatCon$Entrop`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$NormEntropUser`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$NormEntropProd`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$AvNormEntrop`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$GeomAvNormEntrop`
## ------------------------------------------------

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

```
p<-MatCon$new(A,Source="Congalton and Green 2008")
p$GeomAvNormEntrop()


## ------------------------------------------------
## Method `MatCon$AvMaxNormEntrop`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Muñoz 2016")
p$Tau()


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$UserProdAcc()


## ------------------------------------------------
## Method `MatCon$DetailedKappa`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$DetailedCondKappa`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$QES`
```

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

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


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A, Source="Congalton and Green 2008")
p$MTypify(RaR=5)

## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
p$AllParameters()


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A, Source="Congalton and Green 2008")
p$MBootStrap(2)


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Muñoz 2016")
p$MNormalize()$values


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Muñoz 2016")
p$MPseudoZeroes()


## ------------------------------------------------
## Method `MatCon$DetailedWTau`
## ------------------------------------------------

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<-MatCon$new(A,Source="Congalton and Green 2008")
p$DetailedWTau(WV)


## ------------------------------------------------
## Method `MatCon$DetailedWKappa`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$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<-MatCon$new(A,Source="Congalton and Green 2008")
WM<- t(matrix(c(1,0,0.67,1,0,1,0,0,1,0,1,1,0.91,0,0.61,1),
nrow = 4, ncol=4))
p$UserProdAcc_W(WM)


## ------------------------------------------------
## Method `MatCon$StHell`
## ------------------------------------------------

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


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


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


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


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


## ------------------------------------------------
## Method `MatCon$plot.global`
## ------------------------------------------------

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


## ------------------------------------------------
## Method `MatCon$plot.class`
## ------------------------------------------------

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

```
p<-MatCon$new(A,Source="Congalton and Green 2008")
p$plot.class()
```

QCCS *Quality Control Columns Set*

Description

The p-value is calculated from,the given data and probability vectors, which follow a multinomial or binomial distribution. Hypothesis contrasts are applied and it is decided whether the classification of elements is optimal or not.

Value

Object of class QCCS.

Methods

Public methods:

- QCCS$new()
- QCCS$Exact.test()
- QCCS$JiGlobal.test()
- QCCS$Ji.test()
- QCCS$clone()

Method `new()`**:** Public method to create an instance of the QCCS class. At the time of creation, a list of vectors with data and a list of probability vectors that correspond to the data must be defined. The same number of data vectors as probability vectors must be entered, the pairs of data-probability vectors must have the same size, otherwise an error will be displayed. The optional possibility of adding metadata to the matrix is offered. The values of the data vectors represent the reference categories that will be taken into account.

Usage:

```
QCCS$new(vectors, prob, ID = NULL, Date = NULL, Source = NULL)
```

Arguments:

`vectors` vector list.

`prob` probabilities list.

`ID` Identifier. By default ID is a date in YYYYMMDD format

`Date` Date provided by the user. By default the date provided by the system will be taken.

`Source` Indicates where the matrix comes from (article, project, etc.). By default is NULL.

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")
```

Method `Exact.test()`**:** Public method that, using a list of vectors and their corresponding probabilities, through a multinomial or binomial distribution depending on the size of the vector, calculates the p value using each pair of vector-probability data. The null hypothesis shows that for each category the data set is either well classified or not. The Bonferroni method is used. The references (Ariza-López et al. 2019; Alba-Fernández et al. 2020) is followed for the computations.

Usage:

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

Arguments:

`alpha` significance level. By default alpha=0.05.

Returns: The p value is obtained for each vector, and using the Bonferroni criterion it is decided whether the elements are well classified or not.

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$new(vectors,prob,
Source="Alba-Fernández et al. 2020")
A$Exact.test()
```

Method `JiGlobal.test()`**:** Public method that, using a list of vectors and their corresponding probabilities that follow a binomial or multinomial distribution, calculates the p-value for compliance with all defined specifications. The chi square test is used. The null hypothesis verifies that the probabilities are met and therefore that the set of elements are well defined. If one of the defined probabilities is not met, the null hypothesis would be rejected. The references (Ariza-López et al. 2019; Alba-Fernández et al. 2020) is followed for the computations.

Usage:

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

Arguments:

`alpha` significance level. By default alpha=0.05.

Returns: The p value of the entire data set is obtained, through the chi-square, and it is decided whether the elements are well classified or not.

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()
```

Method `Ji.test()`**:** Public method that, using a list of vectors and their corresponding probabilities that follow a multinomial or binomial distribution, calculates the p value using each vector-probability data pair. The chi square test is used. The null hypothesis shows that for each category the data set is either well classified or not. The Bonferroni method is used. The references (Ariza-López et al. 2019; Alba-Fernández et al. 2020) is followed for the computations.

Usage:

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

Arguments:

`alpha` significance level. By default alpha=0.05.

Returns: The p value is obtained for each vector, and using the Bonferroni criterion it is decided whether the elements are well classified or not.

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()
```

Method `clone()`: The objects of this class are cloneable with this method.

Usage:

```
QCCS$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Note

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 the elements of the probability vectors is 0.
- `Error type 5`: Some element of the data vector is negative.
- `Error type 6`: Some element of the probability vector is negative.

References

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.

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.

Examples

```
## ------------------------------------------------
## Method `QCCS$new`
## ------------------------------------------------

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")


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

```
## Method `QCCS$Exact.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$new(vectors,prob,
Source="Alba-Fernández et al. 2020")
A$Exact.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()


## ------------------------------------------------
## 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()
```

Index