Package ‘ConsRank’

December 5, 2019

Type Package

Title Compute the Median Ranking(s) According to the Kemeny's Axiomatic Approach

Version 2.1.0

Date 2019-12-05

Maintainer Antonio D'Ambrosio <antdambr@unina.it>

Depends rgl

Imports rlist (>= 0.4.2), methods, proxy, gtools

Description Compute the median ranking according to the Kemeny's axiomatic approach.
Rankings can or cannot contain ties, rankings can be both complete or incomplete.
The package contains both branch-and-
bound algorithms and heuristic solutions recently proposed.
The searching space of the solution can either be restricted to the universe of the permutations or unrestricted to all possible ties.
The package also provide some useful utilities for deal with preference rankings.
This release declare as deprecated some functions that are still in the package for compatibility. Next release will not contains these functions.
Please type '?ConsRank-deprecated'
Essential references:
Emond, E.J., and Mason, D.W. (2002) <doi:10.1002/mcda.313>;
D'Ambrosio, A., Amodio, S., and Iorio, C. (2015) <doi:10.1285/i20705948v8n2p198>;
Amodio, S., D'Ambrosio, A., and Siciliano R. (2016) <doi:10.1016/j.ejor.2015.08.048>;
D'Ambrosio, A., Mazzeo, G., Iorio, C., and Siciliano, R. (2017) <doi:10.1016/j.cor.2017.01.017>.

License GPL-3

Encoding UTF-8

URL https://www.r-project.org/

Repository CRAN

RoxygenNote 7.0.1

NeedsCompilation no

Author Antonio D'Ambrosio [aut, cre],
Sonia Amodio [ctb],
Giulio Mazzeo [ctb]

Date/Publication 2019-12-05 19:50:02 UTC

R topics documented:

`ConsRank-package` *Median Ranking Approach According to the Kemeny's Axiomatic Approach*

Description

Compute the median ranking according to the Kemeny's axiomatic approach. Rankings can or cannot contain ties, rankings can be both complete or incomplete. The package contains both branch-and-bound and heuristic solutions as well as routines for computing the median constrained bucket order and the K-median cluster component analysis. The package also contains routines for visualize rankings and for detecting the universe of rankings including ties.

Details

Package:	ConsRank
Type:	Package
Version:	2.1.0
Date:	2017-04-28
License:	GPL-3

Author(s)

Antonio D'Ambrosio [cre,aut] <antdambr@unina.it>, Sonia Amdio <sonia.amodio@unina.it> [ctb], Giulio Mazzeo [ctb] <giuliomazzeo@gmail.com>

Maintainer: Antonio D'Ambrosio <antdambr@unina.it>

References

Kemeny, J. G., & Snell, J. L. (1962). Mathematical models in the social sciences (Vol. 9). New York: Ginn.

Marden, J. I. (1996). Analyzing and modeling rank data. CRC Press.

Emond, E. J., & Mason, D. W. (2002). A new rank correlation coefficient with application to the consensus ranking problem. Journal of Multi-Criteria Decision Analysis, 11(1), 17-28.

D'Ambrosio, A. (2008). Tree based methods for data editing and preference rankings. Ph.D. thesis. http://www.fedoa.unina.it/id/eprint/2746

Heiser, W. J., & D'Ambrosio, A. (2013). Clustering and prediction of rankings within a Kemeny distance framework. In Algorithms from and for Nature and Life (pp. 19-31). Springer International Publishing.

Amodio, S., D'Ambrosio, A. & Siciliano, R (2016). Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach. European Journal of Operational Research, vol. 249(2).

D'Ambrosio, A., Amodio, S. & Iorio, C. (2015). Two algorithms for finding optimal solutions of the Kemeny rank aggregation problem for full rankings. Electronic Journal of Applied Statistical Analysis, vol. 8(2).

D'Ambrosio, A., Mazzeo, G., Iorio, C., & Siciliano, R. (2017). A differential evolution algorithm for finding the median ranking under the Kemeny axiomatic approach. Computers & Operations Research, vol. 82.

D'Ambrosio, A., & Heiser, W.J. (2019). A Distribution-free Soft Clustering Method for Preference Rankings. Behaviormetrika , vol. 46(2), pp. 333–351.

D'Ambrosio, A., Iorio, C., Staiano, M., & Siciliano, R. (2019). Median constrained bucket order rank aggregation. Computational Statitstics, vol. 34(2), pp. 787–802,

Examples

```
## load APA data set, full version
data(APAFULL)
## Emond and Mason Branch-and-Bound algorithm.
#CR=consrank(APAFULL)
#use frequency tables
#TR=tabulaterows(APAFULL)
#quick algorithm
#CR2=consrank(TR$X,wk=TR$Wk,algorithm="quick")
#FAST algorithm
#CR3=consrank(TR$X,wk=TR$Wk,algorithm="fast",itermax=10)
#Decor algorithm
#CR4=consrank(TR$X,wk=TR$Wk,algorithm="decor",itermax=10)

#####################################
### load sports data set
#data(sports)
### FAST algorithm
#CR=consrank(sports,algorithm="fast",itermax=10)
#####################################

#######################################
### load Emond and Mason data set
#data(EMD)
### matrix X contains rankings
#X=EMD[,1:15]
### vector Wk contains frequencies
#Wk=EMD[,16]
### QUICK algorithm
#CR=consrank(X,wk=Wk,algorithm="quick")
#######################################
```

APAFULL *American Psychological Association dataset, full version*

Description

The American Psychological Association dataset includes 15449 ballots of the election of the president in 1980, 5738 of which are complete rankings, in which the candidates are ranked from most to least favorite.

Usage

```
data(APAFULL)
```

Source

Diaconis, P. (1988). Group representations in probability and statistics. Lecture Notes-Monograph Series, i-192., pag. 96.

`APAred` *American Psychological Association dataset, reduced version with only full rankings*

Description

The American Psychological Association reduced dataset includes 5738 ballots of the election of the president in 1980, in which the candidates are ranked from most to least favorite.

Usage

```
data(APAred)
```

Source

Diaconis, P. (1988). Group representations in probability and statistics. Lecture Notes-Monograph Series, i-192., pag. 96.

`BBFULL` *Branch-and-Bound algorithm to find the median ranking in the space of full (or complete) rankings.*

Description

Branch-and-bound algorithm to find consensus ranking as definned by D'Ambrosio et al. (2015). If the number of objects to be ranked is large (greater than 20 or 25), it can work for very long time. Use either QuickCons or FASTcons with the option FULL=TRUE instead

Usage

```
BBFULL(X, Wk = NULL, PS = TRUE)
```

Arguments

`X`	A N by M data matrix, in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. The data matrix can contain both full and tied rankings, or incomplete rankings. Alternatively X can contain the rankings observed only once. In this case the argument Wk must be used
`Wk`	Optional: the frequency of each ranking in the data
`PS`	If PS=TRUE, on the screen some information about how many branches are processed are displayed

Details

This function is deprecated and it will be removed in the next release of the package. Use function 'consrank' instead.

If the objects to be ranked is large (>25 - 30), it can take long time to find the solutions

Value

a "list" containing the following components:

Consensus	the Consensus Ranking
Tau	averaged TauX rank correlation coefficient
Eltime	Elapsed time in seconds

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

D'Ambrosio, A., Amodio, S., and Iorio, C. (2015). Two algorithms for finding optimal solutions of the Kemeny rank aggregation problem for full rankings. Electronic Journal of Applied Statistical Analysis, 8(2), 198-213.

See Also

consrank

Examples

```
#data(APAFULL)
#CR=BBFULL(APAFULL)
```

BU *Brook and Upton data*

Description

The data consist of ballots of three candidates, where the 948 voters rank the candidates from 1 to 3. Data are in form of frequency table.

Usage

```
data(BU)
```

Source

Brook, D., & Upton, G. J. G. (1974). Biases in local government elections due to position on the ballot paper. Applied Statistics, 414-419.

References

Marden, J. I. (1996). Analyzing and modeling rank data. CRC Press, pag. 153.

Examples

```
data(BU)
polyplot(BU[,1:3],Wk=BU[,4])
```

`combinpmatr` *Combined input matrix of a data set*

Description

Compute the Combined input matrix of a data set as defined by Emond and Mason (2002)

Usage

```
combinpmatr(X, Wk = NULL)
```

Arguments

`X`	A data matrix N by M, in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. Alternatively X can contain the rankings observed only once. In this case the argument Wk must be used
`Wk`	Optional: the frequency of each ranking in the data

Value

The M by M combined input matrix

Author(s)

Antonio D'Ambrosio `<antdambr@unina.it>`

References

Emond, E. J., and Mason, D. W. (2002). A new rank correlation coefficient with application to the consensus ranking problem. Journal of Multi-Criteria Decision Analysis, 11(1), 17-28.

See Also

`tabulaterows` frequency distribution of a ranking data.

Examples

```
data(APAred)
CI<-combinpmatr(APAred)
TR<-tabulaterows(APAred)
CI<-combinpmatr(TR$X,TR$Wk)
```

---

consrank — *Branch-and-bound and heuristic algorithms to find consensus (median) ranking according to the Kemeny's axiomatic approach*

Description

Branch-and-bound, Quick , FAST and DECOR algorithms to find consensus (median) ranking according to the Kemeny's axiomatic approach. The median ranking(s) can be restricted to be necessarily a full ranking, namely without ties

Usage

```
consrank(
  X,
  wk = NULL,
  ps = TRUE,
  algorithm = "BB",
  full = FALSE,
  itermax = 10,
  np = 15,
  gl = 100,
  ff = 0.4,
  cr = 0.9,
  proc = FALSE
)
```

Arguments

`X`	A n by m data matrix, in which there are n judges and m objects to be judged. Each row is a ranking of the objects which are represented by the columns. If X contains the rankings observed only once, the argument wk can be used
`wk`	Optional: the frequency of each ranking in the data
`ps`	If PS=TRUE, on the screen some information about how many branches are processed are displayed.
`algorithm`	Specifies the used algorithm. One among "BB", "Quick", "FAST" and "DECOR". algorithm="BB" is the default option.
`full`	Specifies if the median ranking must be searched in the universe of rankings including all the possible ties (full=FALSE) or in the restricted space of full rankings (permutations). full=FALSE is the default option.

`itermax`	maximum number of iterations for FAST and DECOR algorithms. itermax=10 is the default option.
`np`	For DECOR algorithm only: the number of population individuals. np=15 is the default option.
`gl`	For DECOR algorithm only: generations limit, maximum number of consecutive generations without improvement. gl=100 is the default option.
`ff`	For DECOR algorithm only: the scaling rate for mutation. Must be in [0,1]. ff=0.4 is the default option.
`cr`	For DECOR algorithm only: the crossover range. Must be in [0,1]. cr=0.9 is the default option.
`proc`	For BB algorithm only: proc=TRUE allows the branch and bound algorithm to work in difficult cases, i.e. when the number of objects is larger than 15 or 25. proc=FALSE is the default option

Details

The BB algorithm can take long time to find the solutions if the number objects to be ranked is large with some missing (>15-20 if full=FALSE, <25-30 if full=TRUE). quick algorithm works with a large number of items to be ranked. The solution is quite accurate. fast algorithm works with a large number of items to be ranked by repeating several times the quick algorithm with different random starting points. decor algorithm works with a very large number of items to be ranked. For decor algorithm, empirical evidence shows that the number of population individuals (the 'np' parameter) can be set equal to 10, 20 or 30 for problems till 20, 50 and 100 items. Both scaling rate and crossover ratio (parameters 'ff' and 'cr') must be set by the user. The default options (ff=0.4, cr=0.9) work well for a large variety of data sets All algorithms allow the user to set the option 'full=TRUE' if the median ranking(s) must be searched in the restricted space of permutations instead of in the unconstrained universe of rankings of n items including all possible ties

Value

a "list" containing the following components:

Consensus	the Consensus Ranking
Tau	averaged TauX rank correlation coefficient
Eltime	Elapsed time in seconds

#'

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

Emond, E. J., and Mason, D. W. (2002). A new rank correlation coefficient with application to the consensus ranking problem. Journal of Multi-Criteria Decision Analysis, 11(1), 17-28. D'Ambrosio, A., Amodio, S., and Iorio, C. (2015). Two algorithms for finding optimal solutions of the Kemeny rank aggregation problem for full rankings. Electronic Journal of Applied Statistical Analysis, 8(2), 198-213. Amodio, S., D'Ambrosio, A. and Siciliano, R. (2016). Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach. European Journal of Operational Research, 249(2), 667-676. D'Ambrosio, A., Mazzeo, G., Iorio, C., and Siciliano, R. (2017). A differential evolution algorithm for finding the median ranking under the Kemeny axiomatic approach. Computers and Operations Research, vol. 82, pp. 126-138.

Examples

```
data(Idea)
RevIdea<-6-Idea
# as 5 means "most associated", it is necessary compute the reverse ranking of
# each rankings to have rank 1 = "most associated" and rank 5 = "least associated"
CR<-consrank(RevIdea)
CR<-consrank(RevIdea,algorithm="quick")
#CR<-consrank(RevIdea,algorithm="fast",itermax=10)
#not run
#data(EMD)
#CRemd<-consrank(EMD[,1:15],wk=EMD[,16],algorithm="decor",itermax=1)
#data(APAFULL)
#CRapa<-consrank(APAFULL,full=TRUE)
```

ConsRank-deprecated *Deprecated functions in ConsRank*

Description

These functions still work but will be removed (defunct) in the next version.

Details

- EMCons;
- QuickCons;
- BBFULL;
- FASTcons;
- DECOR;
- FASTDECOR;
- labels;

All these functions are deprecated, and will be removed in the next release of this package. The functions still remain in the package for compatibility of ConsRank users

See Also

consrank

rank2order

DECOR *Differential Evolution algorithm for Median Ranking*

Description

Differential evolution algorithm for median ranking detection. It works with full, tied and partial rankings. The solution con be constrained to be a full ranking or a tied ranking

Usage

```
DECOR(X, Wk = NULL, NP = 15, L = 100, FF = 0.4, CR = 0.9, FULL = FALSE)
```

Arguments

`X`	A N by M data matrix, in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. Alternatively X can contain the rankings observed only once. In this case the argument Wk must be used
`Wk`	Optional: the frequency of each ranking in the data
`NP`	The number of population individuals
`L`	Generations limit: maximum number of consecutive generations without improvement
`FF`	The scaling rate for mutation. Must be in [0,1]
`CR`	The crossover range. Must be in [0,1]
`FULL`	Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of full rankings.

Details

This function is deprecated and it will be removed in the next release of the package. Use function 'consrank' instead.

Value

a "list" containing the following components:

Consensus	the Consensus Ranking
Tau	averaged TauX rank correlation coefficient
Eltime	Elapsed time in seconds

Author(s)

Antonio D'Ambrosio `<antdambr@unina.it>` and Giulio Mazzeo `<giuliomazzeo@gmail.com>`

References

D'Ambrosio, A., Mazzeo, G., Iorio, C., and Siciliano, R. (2017). A differential evolution algorithm for finding the median ranking under the Kemeny axiomatic approach. Computers and Operations Research, vol. 82, pp. 126-138.

See Also

consrank

Examples

```
#not run
#data(EMD)
#CR=DECOR(EMD[,1:15],EMD[,16])
```

EMCons — *Branch-and-bound algorithm to find consensus (median) ranking according to the Kemeny's axiomatic approach*

Description

Branch-and-bound algorithm to find consensus ranking as definned by Emond and Mason (2002). If the number of objects to be ranked is large (greater than 15 or 20, specially if there are missing rankings), it can work for very long time.

Usage

```
EMCons(X, Wk = NULL, PS = TRUE)
```

Arguments

X	A N by M data matrix, in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. Alternatively X can contain the rankings observed only once. In this case the argument Wk must be used
Wk	Optional: the frequency of each ranking in the data
PS	If PS=TRUE, on the screen some information about how many branches are processed are displayed

Details

This function is deprecated and it will be removed in the next release of the package. Use function 'consrank' instead.

Value

a "list" containing the following components:

Consensus	the Consensus Ranking
Tau	averaged TauX rank correlation coefficient
Eltime	Elapsed time in seconds

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

Emond, E. J., and Mason, D. W. (2002). A new rank correlation coefficient with application to the consensus ranking problem. Journal of Multi-Criteria Decision Analysis, 11(1), 17-28.

See Also

consrank

Examples

```
data(Idea)
RevIdea=6-Idea
# as 5 means "most associated", it is necessary compute the reverse ranking of
# each rankings to have rank 1 = "most associated" and rank 5 = "least associated"
CR=EMCons(RevIdea)
```

EMD *Emond and Mason data*

Description

Data simuated by Emond and Mason to check their branch-and-bound algorithm. There are 112 voters ranking 15 objects. There are 21 uncomplete rankings. Data are in form of frequency table.

Usage

```
data(EMD)
```

Source

Emond, E. J., & Mason, D. W. (2000). A new technique for high level decision support. Department of National Defence, Operational Research Division, pag. 28.

References

Emond, E. J., & Mason, D. W. (2000). A new technique for high level decision support. Department of National Defence, Operational Research Division, pag. 28.

Examples

```
data(EMD)
CR=consrank(EMD[,1:15],EMD[,16],algorithm="quick")
```

FASTcons *FAST algorithm to find consensus (median) ranking. FAST algorithm to find consensus (median) ranking defined by Amodio, D'Ambrosio and Siciliano (2016). It returns at least one of the solutions. If there are multiple solutions, sometimes it returns all the solutions, sometimes it returns some solutions, always it returns at least one solution.*

Description

FAST algorithm to find consensus (median) ranking.

FAST algorithm to find consensus (median) ranking defined by Amodio, D'Ambrosio and Siciliano (2016). It returns at least one of the solutions. If there are multiple solutions, sometimes it returns all the solutions, sometimes it returns some solutions, always it returns at least one solution.

Usage

```
FASTcons(X, Wk = NULL, maxiter = 50, FULL = FALSE, PS = FALSE)
```

Arguments

`X`	is a ranking data matrix
`Wk`	is a vector of weights
`maxiter`	maximum number of iterations: default = 50.
`FULL`	Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of full rankings.
`PS`	Default PS=FALSE. If PS=TRUE the number of current iteration is diplayed

Details

This function is deprecated and it will be removed in the next release of the package. Use function 'consrank' instead.

Value

a "list" containing the following components:

Consensus	the Consensus Ranking
Tau	averaged TauX rank correlation coefficient
Eltime	Elapsed time in seconds

Author(s)

Antonio D'Ambrosio <antdambr@unina.it> and Sonia Amodio <sonia.amodio@unina.it>

References

Amodio, S., D'Ambrosio, A. and Siciliano, R. (2016). Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach. European Journal of Operational Research, 249(2), 667-676.

See Also

EMCons Emond and Mason branch-and-bound algorithm.

QuickCons Quick algorithm.

Examples

```
##data(EMD)
##X=EMD[,1:15]
##Wk=matrix(EMD[,16],nrow=nrow(X))
##CR=FASTcons(X,Wk,maxiter=100)
##These lines produce all the three solutions in less than a minute.

data(sports)
CR=FASTcons(sports,maxiter=5)
```

FASTDECOR *FAST algorithm calling DECOR*

Description

FAST algorithm repeats DECOR a prespecified number of time. It returns the best solutions among the iterations

Usage

```
FASTDECOR(
  X,
  Wk = NULL,
  maxiter = 10,
  NP = 15,
  L = 100,
  FF = 0.4,
  CR = 0.9,
  FULL = FALSE,
  PS = TRUE
)
```

Arguments

X	A N by M data matrix, in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. Alternatively X can contain the rankings observed only once. In this case the argument Wk must be used
Wk	Optional: the frequency of each ranking in the data
maxiter	maximum number of iterations. Default 10
NP	The number of population individuals
L	Generations limit: maximum number of consecutive generations without improvement
FF	The scaling rate for mutation. Must be in [0,1]
CR	The crossover range. Must be in [0,1]
FULL	Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of full rankings. In this case, the data matrix must contain full rankings.
PS	Default PS=TRUE. If PS=TRUE the number of a multiple of 5 iterations is diplayed

Details

This function is deprecated and it will be removed in the next release of the package. Use function 'consrank' instead.

Value

a "list" containing the following components:

Consensus	the Consensus Ranking
Tau	averaged TauX rank correlation coefficient
Eltime	Elapsed time in seconds

Author(s)

Antonio D'Ambrosio <antdambr@unina.it> and Giulio Mazzeo <giuliomazzeo@gmail.com>

References

D'Ambrosio, A., Mazzeo, G., Iorio, C., and Siciliano, R. (2017). A differential evolution algorithm for finding the median ranking under the Kemeny axiomatic approach. Computers and Operations Research, vol. 82, pp. 126-138.

See Also

consrank

Examples

```
#data(EMD)
```

```
#CR=FASTDECOR(EMD[,1:15],EMD[,16])
```

German *German political goals*

Description

Ranking data of 2262 German respondents about the desirability of the four political goals: a = the maintenance of order in the nation; b = giving people more say in the decisions of government; c = growthing rising prices; d = protecting freedom of speech

Usage

```
data(German)
```

Source

Croon, M. A. (1989). Latent class models for the analysis of rankings. Advances in psychology, 60, 99-121.

Examples

```
data(German)
TR=tabulaterows(German)
polyplot(TR$X,Wk=TR$Wk,nobj=4)
```

Idea *Idea data set*

Description

98 college students where asked to rank five words, (thought, play, theory, dream, attention) regarding its association with the word idea, from 5=most associated to 1=least associated.

Usage

```
data(Idea)
```

Source

Fligner, M. A., & Verducci, J. S. (1986). Distance based ranking models. Journal of the Royal Statistical Society. Series B (Methodological), 359-369.

Examples

```
data(Idea)
revIdea=6-Idea
TR=tabulaterows(revIdea)
CR=consrank(TR$X,wk=TR$Wk,algorithm="quick")
colnames(CR$Consensus)=colnames(Idea)
```

---

kemenyd *Kemeny distance*

Description

Compute the Kemeny distance of a data matrix containing preference rankings, or compute the kemeny distance between two (matrices containing) rankings.

Usage

```
kemenyd(X, Y = NULL)
```

Arguments

X	A N by M data matrix, in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. If there is only X as input, the output is a square distance matrix
Y	A row vector, or a n by M data matrix in which there are n judges and the same M objects as X to be judged.

Value

If there is only X as input, d = square distance matrix. If there is also Y as input, d = matrix with N rows and n columns.

Author(s)

Antonio D'Ambrosio `<antdambr@unina.it>`

References

Kemeny, J. G., & Snell, L. J. (1962). Preference ranking: an axiomatic approach. Mathematical models in the social sciences, 9-23.

See Also

`tau_x` TauX rank correlation coefficient

Examples

```
data(Idea)
RevIdea<-6-Idea ##as 5 means "most associated", it is necessary compute the reverse
#ranking of each rankings to have rank 1 = "most associated" and rank 5 = "least associated"
KD<-kemenyd(RevIdea)
KD2<-kemenyd(RevIdea[1:10,],RevIdea[55,])
```

---

kemenydesign *Auxiliary function*

Description

Define a design matrix to compute Kemeny distance

Usage

```
kemenydesign(X)
```

Arguments

X	A N by M data matrix, in which there are N judges and M objects to be judged. Each row is a ranking of the objects represented by the columns.

Value

Design matrix

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

D'Ambrosio, A. (2008). Tree based methods for data editing and preference rankings. Unpublished PhD Thesis. Universita' degli Studi di Napoli Federico II.

kemenyscore *Score matrix according Kemeny (1962)*

Description

Given a ranking, it computes the score matrix as defined by Emond and Mason (2002)

Usage

```
kemenyscore(X)
```

Arguments

X	a ranking (must be a row vector or, better, a matrix with one row and M columns)

Value

the M by M score matrix

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

Kemeny, J and Snell, L. (1962). Mathematical models in the social sciences.

See Also

scorematrix The score matrix as defined by Emond and Mason (2002)

Examples

```
Y <- matrix(c(1,3,5,4,2),1,5)
SM<-kemenyscore(Y)
#
Z<-c(1,2,3,2)
SM2<-kemenyscore(Z)
```

labels — *Transform a ranking into a ordering.*

Description

Given a ranking (or a matrix of rank data), transforms it into an ordering (or a ordering matrix)

Usage

```
labels(x, m, label = 1:m, labs)
```

Arguments

`x`	a ranking, or a n by m data matrix in which there are n judges ranking m objects
`m`	the number of objects
`label`	optional: the name of the objects
`labs`	labs = 1 displays the names of the objects if there is argument "label", otherwise displays the permutation of first m integer. labs = 2 is to be used only if the argument "label" is not defined. In such a case it displays the permutation of the first m letters

Details

This function is deprecated and it will be removed in the next release of the package. Use function 'rank2order' instead.

Value

the ordering

Author(s)

Sonia Amodio <sonia.amodio@unina.it>

See Also

rank2order

Examples

```
data(Idea)
TR=tabulaterows(Idea)
Ord=labels(TR$X,ncol(Idea),colnames(Idea),labs=1)
Ord2=labels(TR$X,ncol(Idea),labs=2)
cbind(Ord,TR$Wk)
cbind(Ord2,TR$Wk)
```

order2rank — *Given an ordering, it is transformed to a ranking*

Description

From ordering to rank. IMPORTANT: check which symbol denotes tied rankings in the X matrix

Usage

```
order2rank(X, TO = "{", TC = "}")
```

Arguments

X	A ordering or a matrix containing orderings
TO	symbol indicating the start of a set of items ranked in a tie
TC	symbol indicating the end of a set of items ranked in a tie

Value

a ranking or a matrix of rankings:

R	ranking or matrix of rankings

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

Examples

```
data(APAred)
ord=rank2order(APAred) #transform rankings into orderings
ran=order2rank(ord) #transform the orderings into rankings
```

partitions — *Generate partitions of n items constrained into k non empty subsets*

Description

Generate all possible partitions of n items constrained into k non empty subsets. It does not generate the universe of rankings constrained into k buckets.

Usage

```
partitions(n, k = NULL, items = NULL, itemtype = "L")
```

Arguments

n	a (integer) number denoting the number of items
k	The number of the non-empty subsets. Default value is NULL, in this case all the possible partitions are displayed
items	items: the items to be placed into the ordering matrix. Default are the first c small letters
itemtype	to be used only if items is not set. The default value is "L", namely letters. Any other symbol produces items as the first c integers

Details

If the objects to be ranked is large (>15-20) with some missing, it can take long time to find the solutions. If the searching space is limited to the space of full rankings (also incomplete rankings, but without ties), use the function BBFULL or the functions FASTcons and QuickCons with the option FULL=TRUE.

Value

the ordering matrix (or vector)

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

See Also

stirling2 Stirling number of second kind.

rank2order Convert rankings into orderings.

order2rank Convert orderings into ranks.

univranks Generate the universe of rankings given the input partition

Examples

```
X<-partitions(4,3)
#shows all the ways to partition 4 items (say "a", "b", "c" and "d" into 3 non-empty subets
 #(i.e., into 3 buckets). The Stirling number of the second kind (4,3) indicates that there
 #are 6 ways.
s2<-stirling2(4,3)$S
X2<-order2rank(X) #it transform the ordering into ranking
```

polyplot *Plot rankings on a permutation polytope of 3 o 4 objects containing all possible ties*

Description

Plot rankings a permutation polytope that is the geometrical space of preference rankings. The plot is available for 3 or for 4 objects

Usage

```
polyplot(X = NULL, L = NULL, Wk = NULL, nobj = 3)
```

Arguments

X	the sample of rankings. Most of the time it is returned by tabulaterows
L	labels of the objects
Wk	frequency associated to each ranking
nobj	number of objects. It must be either 3 or 4

Details

polyplot() plots the universe of 3 objecys. polyplot(nobj=4) plots the universe of 4 objecys.

Value

the permutation polytope

Author(s)

Antonio D'Ambrosio <antdambr@unina.it> and Sonia Amodio <sonia.amodio@unina.it>

References

Thompson, G. L. (1993). Generalized permutation polytopes and exploratory graphical methods for ranked data. The Annals of Statistics, 1401-1430. # Heiser, W. J., and D'Ambrosio, A. (2013). Clustering and prediction of rankings within a Kemeny distance framework. In Algorithms from and for Nature and Life (pp. 19-31). Springer International Publishing.

See Also

tabulaterows frequency distribution for ranking data.

Examples

```
polyplot()
#polyplot(nobj=4)
data(BU)
polyplot(BU[,1:3],Wk=BU[,4])
```

QuickCons — *Quick algorithm to find up to 4 solutions to the consensus ranking problem*

Description

The Quick algorithm finds up to 4 solutions. Solutions reached are most of the time optimal solutions.

Usage

```
QuickCons(X, Wk = NULL, FULL = FALSE, PS = FALSE)
```

Arguments

`X`	A N by M data matrix in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. Alternatively X can contain the rankings observed only once in the sample. In this case the argument Wk must be used
`Wk`	Optional: the frequency of each ranking in the data
`FULL`	Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of full rankings.
`PS`	Default PS=FALSE. If PS=TRUE the number of evaluated branches is diplayed

Details

This function is deprecated and it will be removed in the next release of the package. Use function 'consrank' instead.

Value

a "list" containing the following components:

Consensus	the Consensus Ranking
Tau	averaged TauX rank correlation coefficient
Eltime	Elapsed time in seconds

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

Amodio, S., D'Ambrosio, A. and Siciliano, R. (2016). Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach. European Journal of Operational Research, 249(2), 667-676.

See Also

`consrank`

Examples

```
data(EMD)
CR=QuickCons(EMD[,1:15],EMD[,16])
```

`rank2order` *Given a rank, it is transformed to a ordering*

Description

From ranking to ordering. IMPORTANT: check which symbol denotes tied rankings in the X matrix

Usage

```
rank2order(X, items = NULL, TO = "{", TC = "}", itemtype = "L")
```

Arguments

`X`	A ordering or a matrix containing orderings
`items`	items to be placed into the ordering matrix. Default are the
`TO`	symbol indicating the start of a set of items ranked in a tie
`TC`	symbol indicating the end of a set of items ranked in a tie
`itemtype`	to be used only if items=NULL. The default value is "L", namely

Value

a ordering or a matrix of orderings:

out	ranking or matrix of rankings

Author(s)

Antonio D'Ambrosio `<antdambr@unina.it>`

Examples

```
data(APAred)
```

```
ord<-rank2order(APAred)
```

`reordering` *Given a vector (or a matrix), returns an ordered vector (or a matrix with ordered vectors)*

Description

Given a ranking of M objects (or a matrix with M columns), it reduces it in "natural" form (i.e., with integers from 1 to M)

Usage

```
reordering(X)
```

Arguments

`X`	a ranking, or a ranking data matrix

Value

a ranking in natural form

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

`scorematrix` *Score matrix according Emond and Mason (2002)*

Description

Given a ranking, it computes the score matrix as defined by Emond and Mason (2002)

Usage

```
scorematrix(X)
```

Arguments

`X`	a ranking (must be a row vector or, better, a matrix with one row and M columns)

Value

the M by M score matrix

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

Emond, E. J., and Mason, D. W. (2002). A new rank correlation coefficient with application to the consensus ranking problem. Journal of Multi-Criteria Decision Analysis, 11(1), 17-28.

See Also

`combinpmatr` The combined inut matrix

Examples

```
Y <- matrix(c(1,3,5,4,2),1,5)
SM<-scorematrix(Y)
#
Z<-c(1,2,4,3)
SM2<-scorematrix(Z)
```

`sports` *sports data*

Description

130 students at the University of Illinois ranked seven sports according to their preference (Baseball, Football, Basketball, Tennis, Cycling, Swimming, Jogging).

Usage

```
data(sports)
```

Source

Marden, J. I. (1996). Analyzing and modeling rank data. CRC Press.

Examples

```
data(sports)
```

stirling2 — *Stirling numbers of the second kind*

Description

Denote the number of ways to partition a set of n objects into k non-empty subsets

Usage

```
stirling2(n, k)
```

Arguments

n	(integer): the number of the objects
k	(integer <=n): the number of the non-empty subsets (buckets)

Value

a "list" containing the following components:

S	the stirling number of the second kind
SM	a matrix showing, for each k (on the columns) in how many ways the n objects (on the rows) can be partitioned

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

Comtet, L. (1974). Advanced Combinatorics: The art of finite and infinite expansions. D. Reidel, Dordrecth, The Netherlands.

Examples

```
parts<-stirling2(4,2)
```

tabulaterows — *Frequency distribution of a sample of rankings*

Description

Given a sample of preference rankings, it compute the frequency associated to each ranking

Usage

```
tabulaterows(X, miss = FALSE)
```

Arguments

X	a N by M data matrix containing N judges judging M objects
miss	TRUE if there are missing data (either partial or incomplete rankings): default: FALSE

Value

a "list" containing the following components:

X	the unique rankings
Wk	the frequency associated to each ranking
tabfreq	frequency table

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

Examples

```
data(Idea)
TR<-tabulaterows(Idea)
FR<-TR$Wk/sum(TR$Wk)
RF<-cbind(TR$X,FR)
colnames(RF)<-c(colnames(Idea),"fi")
#compute modal ranking
maxfreq<-which(RF[,6]==max(RF[,6]))
rank2order(RF[maxfreq,1:5],items=colnames(Idea))
#
data(APAred)
TR<-tabulaterows(APAred)
#
data(APAFULL)
TR<-tabulaterows(APAFULL)
CR1<-consrank(TR$X,wk=TR$Wk)
CR2<-consrank(TR$X,wk=TR$Wk,algorithm="fast",itermax=15)
CR3<-consrank(TR$X,wk=TR$Wk,algorithm="quick")
```

---

tau_x — *TauX (tau exstension) rank correlation coefficient*

Description

Tau exstension is a new rank correlation coefficient defined by Emond and Mason (2002)

Usage

```
tau_x(X, Y = NULL)

Tau_X(X, Y = NULL)
```

Arguments

X — a M by N data matrix, in which there are N judges and M objects to be judged. Each row is a ranking of the objects which are represented by the columns. If there is only X as input, the output is a square matrix containing the Tau_X rcc.

Y — A row vector, or a n by M data matrix in which there are n judges and the same M objects as X to be judged.

Value

Tau_x rank correlation coefficient

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References

Emond, E. J., and Mason, D. W. (2002). A new rank correlation coefficient with application to the consensus ranking problem. Journal of Multi-Criteria Decision Analysis, 11(1), 17-28.

See Also

kemenyd Kemeny distance

Examples

```
data(BU)
RD<-BU[,1:3]
Tau<-tau_x(RD)
Tau1_3<-tau_x(RD[1,],RD[3,])
```

---

univranks — *Generate the universe of rankings*

Description

Generate the universe of rankings given the input partition

Usage

```
univranks(X, k = NULL, ordering = TRUE)
```

Arguments

`X`	A ranking, an ordering, a matrix of rankings, a matrix of orderings or a number
`k`	Optional: the number of the non-empty subsets. It has to be used only if X is anumber. The default value is NULL, In this case the universe of rankings with n=X items are computed
`ordering`	The universe of rankings must be returned as orderings (default) or rankings?

Details

The function should be used with small numbers because it can generate a large number of permutations. The use of X greater than 9, of X matrices with more than 9 columns as input is not reccomended.

Value

a "list" containing the following components:

Runiv		The universe of rankings
Cuniv		A list containing:
	R	The universe of rankings in terms of rankings;
	Parts	for each ranking in input the produced rankings
	Univinbuckets	the universe of rankings within each bucket

Author(s)

Antonio D'Ambrosio `<antdambr@unina.it>`

See Also

`stirling2` Stirling number of second kind.

`rank2order` Convert rankings into orderings.

`order2rank` Convert orderings into ranks.

`partitions` Generate partitions of n items constrained into k non empty subsets.

Examples

```
S2<-stirling2(4,4)$SM[4,] #indicates in how many ways 4 objects
                         #can be placed, respectively, into 1, 2,
                         #3 or 4 non-empty subsets.
CardConstr<-factorial(c(1,2,3,4))*S2  #the cardinality of rankings
                                     #constrained into 1, 2, 3 and 4
                                     #buckets
Card<-sum(CardConstr)  #Cardinality of the universe of rankings with 4
                      #objects
U<-univranks(4)$Runiv #the universe of rankings with four objects
                     # we know that the universe counts 75
                     #different rankings
Uk<-univranks(4,2)$Runiv    #the universe of rankings of four objects
```

```
                            #constrained into k=2 buckets, we know they are 14
Up<-univranks(c(1,4,3,1))$Runiv  #the universe of rankings with 4 objects
                                 #for which the first and the fourth item
                                 #are tied
```

USAranks *USA rank data*

Description

Random subset of the rankings collected by O'Leary Morgan and Morgon (2010) on the 50 American States. The 368 number of items (the number of American States) is equal to 50, and the number of rankings is equal to 104. These data concern rankings of the 50 American States on three particular aspects: socio-demographic characteristics, health care expenditures and crime statistics.

Usage

```
data(USAranks)
```

Source

Amodio, S., D'Ambrosio, A. & Siciliano, R (2015). Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the Kemeny axiomatic approach. European Journal of Operational Research. DOI: 10.1016/j.ejor.2015.08.048

References

O'Leary Morgan, K., Morgon, S., (2010). State Rankings 2010: A Statistical view of America; Crime State Ranking 2010: Crime Across America; Health Care State Rankings 2010: Health Care Across America. CQ Press.

Examples

```
data(USAranks)
```

Index