Package ‘ConsRank’

April 29, 2017

Type Package
Title Compute the Median Ranking(s) According to the Kemeny’s Axiomatic Approach
Version 2.0.1
Date 2017-04-28
Author Antonio D'Ambrosio <antdambr@unina.it>, Sonia Amodio <sonia.amdio@unina.it>, Giulio Mazzeo <giuliomazzeo@gmail.com>
Maintainer Antonio D'Ambrosio <antdambr@unina.it>
Depends proxy,rgl,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 package also provide some useful utilities for deal with preference rankings. Essential references:
License GPL-3
URL https://www.r-project.org/
Repository CRAN
RoxygenNote 6.0.1
NeedsCompilation no
Date/Publication 2017-04-28 22:24:06 UTC

R topics documented:

ConsRank-package .......................................................... 2
APAFULL .............................................................. 4
APAred ............................................................. 5
ConsRank-package

Compute the Median Ranking 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.
Details

Package: ConsRank
Type: Package
Version: 2.0.1
Date: 2017-04-28
License: GPL-3

Author(s)
Antonio D’Ambrosio <antdambr@unina.it>, Sonia Amdio <sonia.amdio@unina.it>, Giulio Mazzeo <giuliomazzeo@gmail.com>
Maintainer: Antonio D’Ambrosio <antdambr@unina.it>

References

Examples

```r
## load APA data set, full version
data(APAFULL)
## Emond and Mason Branch-and-Bound algorithm.
## If the number of object is higher than 20, EMCons function may work for several minutes.
## Use either QuickCons, DECOR, FASTcons or FASTDECOR instead
CR=EMCons(APAFULL)
```
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

**APAréd**

*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(APAréd)`

**Source**


---

**BBconsensus**

*Find the first approximation to the consensus ranking. Most of the time the output is a solution, maybe not unique*

**Description**

Find a first approximation to the consensus ranking.

**Usage**

`BBconsensus(RR, cij, FULL = FALSE, PS = FALSE)`

**Arguments**

<table>
<thead>
<tr>
<th>RR</th>
<th>Candidate to be the consensus ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>cij</td>
<td>Combined input matrix of the data set</td>
</tr>
<tr>
<td>FULL</td>
<td>Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of complete rankings. FULL=TRUE if the function is called by BBFULL algorithm.</td>
</tr>
<tr>
<td>PS</td>
<td>Default PS=FALSE. If PS=TRUE the number of evaluated branches is displayed</td>
</tr>
</tbody>
</table>

**Value**

a "list" containing the following components:

- cons  a first approximation of the median ranking
- pen   penalty value
**Author(s)**

Antonio D’Ambrosio <antdambr@unina.it>

**References**


---

**BBconsensus2**

*Core function in computing consensus ranking as defined by Emond and Mason (2002)*

---

**Description**

Core function in computing consensus ranking as defined by Emond and Mason (2002), recalled by EMCons function

**Usage**

BBconsensus2(RR, cij, Po, PS = TRUE, FULL = FALSE)

**Arguments**

- **RR**: A ranking
- **cij**: combined input matrix
- **Po**: current penalty
- **PS**: If PS=true, it prints the evaluating branches
- **FULL**: Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of complete rankings. FULL=TRUE if the function is called by BBFULL algorithm.

**Value**

median ranking

**Author(s)**

Antonio D’Ambrosio <antdambr@unina.it>

**References**


BBFULL

See Also

EMCons Emond and Mason branch-and-bound algorithm
BBFULL D’Ambrosio et al. branch-and-bound algorithm for full rankings

Description

Branch-and-bound algorithm to find the median ranking in the space of full (or complete) rankings.

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

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>
Branches discovery

Usage

```
branches(brR, cij, b, Po, ord, Pb, FULL = FALSE)
```

Arguments

- `brR`: Current processed branche of the BB algorithm
- `cij`: Combined input matrix
- `b`: Other inputs recalled by main functions
- `Po`: Other inputs recalled by main functions
- `ord`: Other inputs recalled by main functions
- `Pb`: Other inputs recalled by main functions
- `FULL`: Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of complete rankings. TRUE=TRUE if the function is called by BBFULL algorithm.

Value

A "list" containing the following components:

- `cR`: ranking belonging to the branche
- `pcR`: penalty of the current ranking

References


See Also

- `fastcons` FAST algorithm algorithm.
- `QuickCons` Quick algorithm.

Examples

```
data(APAFULL)
CR=BBFULL(APAFULL)
```
**BU**

**Author(s)**
Antonio D’Ambrosio <antdambr@unina.it>

**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**

```r
data(BU)
```

**Source**


**References**


**Examples**

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

**childclosint**  

**Transform the vector into ranking for DECOR**

**Description**

Closest integer approach is used: the elements of x are rounded to the closest integer. Then check if any solution exists outside of the bounds (and get it back inside the bounds randomly). Finally repair the solution if repetitions exist.

**Usage**

```r
childclosint(r)
```

**Arguments**

- `r`  
  mutated vector
Value

a valid ranking

Author(s)

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

References


childtie

Auxiliary function

Description

Apply discretization and convert to rank

Usage

childtie(r)

Arguments

r a candidate rank vector

Value

a ranking

Author(s)

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

combincost

**Auxiliary function called by DECORcore**

**Description**
Calculates the sum of distances between a candidate ranking and the data set of rankings.

**Usage**
combincost(ranking, cij, M)

**Arguments**
- `ranking`: the candidate ranking
- `cij`: combined input matrix
- `M`: number of judges

**Value**
a "list" containing the following components:
- `tp`: tauX associated to the ranking
- `cp`: cost associated to the ranking

**Author(s)**
Antonio D’Ambrosio <antdambr@unina.it> and Giulio Mazzeo

---

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

See Also
tabulaterows frequency distribution of a ranking data.

Examples
```r
data(APAred)
CI=combinpmatr(APAred)
TR=tabulaterows(APAred)
CI=combinpmatr(TR$X,TR$Wk)
```

---

crossover

Apply the (binomial) crossover for DE algorithm

Description
Binomial crossover stipulates that crossover will occur on each of the D values in a solution whenever a randomly generated number between 0 and 1 is within the CR range.

Usage
crossover(x, v, CR)

Arguments
- `x` target ranking
- `v` donor individual (mutated x)
- `CR` Crossover range

Value
modified ranking

Author(s)
Antonio D’Ambrosio <antdambr@unina.it> and Giulio Mazzeo <giuliomazzeo@gmail.com>
Description

Differential evolution algorithm for median ranking detection. It works with full, tied and partial rankings. The solution can 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

It works with a very large number of items to be ranked. Empirically, 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.

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


See Also

FASTcons FAST algorithm.
QuickCons Quick algorithm.
EMCons Branch-and-bound algorithm.

Examples

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

<table>
<thead>
<tr>
<th>DECORcore</th>
<th>Differential Evolution algorithm for Median Ranking</th>
</tr>
</thead>
</table>

Description

Core function of the DECOR algorithm

Usage

DECORcore(cij, NJ, NP = 15, L = 50, FF = 0.4, CR = 0.9, FULL = FALSE)

Arguments

cij                combined input matrix
NJ                 the number of judges
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.
**Value**

A "list" containing the following components:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConsR</td>
<td>the Consensus Ranking</td>
</tr>
<tr>
<td>Tau</td>
<td>averaged TauX rank correlation coefficient</td>
</tr>
<tr>
<td>besti</td>
<td>matrix of best individuals for every generation</td>
</tr>
<tr>
<td>bestc</td>
<td>vector of best individuals’ cost for every gen</td>
</tr>
<tr>
<td>bests</td>
<td>vector of best individuals avgTau</td>
</tr>
<tr>
<td>maxavgtau</td>
<td>maximum average tauX Eltime</td>
</tr>
<tr>
<td>Eltime</td>
<td>Elapsed time in seconds</td>
</tr>
</tbody>
</table>

**Author(s)**

Antonio D’Ambrosio <antdambr@unina.it> and Giulio Mazzeo <giuiliomazzeo@gmail.com>

**References**


---

**Description**

Branch-and-bound algorithm to find consensus ranking as defined 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**

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

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


See Also

- **FASTcons**: FAST algorithm algorithm.
- **QuickCons**: Quick algorithm.
- **BBFULL**: Branc-and-bound algorithm for full rankings.

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)
FASTcons

Usage

data(EMD)

Source


References


Examples

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

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 displayed
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.amdio@unina.it>

References


See Also

EMCons  Emond and Mason branch-and-bound algorithm.
QuickCons Quick algorithm.

Examples

```r
#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=10)
```

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 \( W_k \) must be used.

**\( W_k \)**  
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 displayed.

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


Examples

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

**Auxiliary function**

**Description**
Find correct branches in the Branch-and-Bound algorithms

**Usage**
findbranches(R, ord, b, FULL = FALSE)

**Arguments**
- **R**: Candidate to be the consensus ranking
- **ord**: other input values recalled by other routines
- **b**: other input values recalled by other routines
- **FULL**: Default FULL=FALSE. If FULL=TRUE, the searching is limited to the space of complete rankings. TRUE=TRUE if the function is called by BBFULL algorithm.

**Value**
- a candidate to be the median ranking

**Author(s)**
Antonio D’Ambrosio <antdambr@unina.it> and Sonia Amodio <sonia.amodio@unina.it>

findconsensusBB

**Find a first ranking candidate to be the median ranking**

**Description**
Auxiliary function: it finds a first ranking candidate to be the consensus ranking by observing the Combined input matrix

**Usage**
findconsensusBB(cij)

**Arguments**
- **cij**: combined input matrix
**Value**

candidate median ranking

**Author(s)**

Antonio D’Ambrosio <antdambr@unina.it>

---

**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 = growing rising prices; d = protecting freedom of speech

**Usage**

data(German)

**Source**


**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**

Examples

data(Idea)
revIdea=6-Idea
TR=tabulaterows(revIdea)
CR=QuickCons(TR$X, TR$Wk)
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


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

kemenyscore  

_score matrix according Kemeny (1962)_

**Description**

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

**Usage**

```r
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**


**See Also**

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

**Examples**

```r
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 an 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

Value

the ordering

Author(s)

Sonia Amodio <sonia.amodio@unina.it>

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)
Penalty

mutaterand1  Mutation phase

Description
Creates mutation vector v based on the current population X. We use the rand/1/bin system

Usage
mutaterand1(X, FF, i)

Arguments
X  population matrix
FF  scaling factor
i  population index to be ignored

Value
the mutated vector

Author(s)
Antonio D’Ambrosio <antdambr@unina.it> and Giulio Mazzeo <giuliomazzeo@gmail.com>

Penalty  Auxiliary function

Description
Assign a penalty to the branches of the FASTcons and QuickCons algorithms

Usage
Penalty(CR, cij, indice)

Arguments
CR  candidate to be the median ranking
cij  combined input matrix
indice  other input called by other functions

Value
a penalty value
PenaltyBB2

Author(s)

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

References


PenaltyBB2

Auxiliary function

Description

Assign a penalty to the branches of the BB algorithms

Usage

PenaltyBB2(cij, candidate, ord)

Arguments

cij combined input matrix

candidate candidate to be the median ranking

cand ord other input called by other functions

Value

computed penalty

Author(s)

Antonio D’Ambrosio <antdambr@unina.it>
polyplot

Plot rankings on a permutation polytope of 3 or 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.amedio@unina.it>

References


See Also

- tabulaterows frequency distribution for ranking data.
**QuickCons**

**Examples**

```r
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**

```r
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 displayed

**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.
reordering

European Journal of Operational Research, 249(2), 667-676.

See Also

FASTcons FAST algorithm.

QuickCons Quick algorithm.

Examples

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

---

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>
ReorderingBB  

**Auxiliary function**

**Description**

It allows to reorder the objects to be processed in the BB algorithms.

**Usage**

ReorderingBB(RR)

**Arguments**

- **RR**: A ranking

**Value**

A reordered ranking

**Author(s)**

Sonia Amodio <sonia.amodio@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


See Also

combinpmatr The combined inout matrix

Examples

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

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


Examples

data(sports)
**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)**

Sonia Amodio <sonia.amadio@unina.it>

**Examples**

```r
data(Idea)
TR=tabulaterows(Idea)
FR=TR$Wk/sum(TR$Wk)
RF=cbind(TR$X,FR)
colnames(RF)=c(colnames(Idea),"fi")
#compute modal ranking
maxfreq=which(RF[,6]==max(RF[,6]))
labs(RF[maxfreq,1:5],5,table,labels=1)
#
data(APAred)
TR=tabulaterows(APAred)
#
data(APAFULL)
TR=tabulaterows(APAFULL)
CR1=EMCons(TR$X,TR$Wk)
CR2=FASTcons(TR$X,TR$Wk,maxiter=15)
CR3=QuickCons(TR$X,TR$Wk)
```
Description

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

Usage

\texttt{tau\_x(X, Y = NULL)}

Arguments

\begin{itemize}
\item \texttt{X} \hspace{1cm} 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 \texttt{X} as input, the output is a square matrix containing the Tau\_X rcc.
\item \texttt{Y} \hspace{1cm} 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.
\end{itemize}

Value

\texttt{tau\_x rank correlation coefficient}

Author(s)

Antonio D'Ambrosio <antdambr@unina.it>

References


See Also

\texttt{kemeny\textunderscore d} Kemeny distance

Examples

\begin{verbatim}
data(BU)
RD=BU[,1:3]
Tau=tau\_X(RD)
Tau1\_3=tau\_X(RD[1,],RD[3,])
\end{verbatim}
**USAranks**

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


References


Examples

data(USAranks)
Index

*Topic **Branch-and-bound algorithms**
  ConsRank-package, 2
*Topic **Consensus ranking**
  ConsRank-package, 2
*Topic **Consensus**
  EMCons, 15
*Topic **Differential evolution algorithms**
  ConsRank-package, 2
*Topic **Differential**
  DECOR, 13
  FASTDECOR, 18
*Topic **FAST**
  FASTcons, 17
*Topic **Genetic**
  DECOR, 13
*Topic **Kemeny distance**
  ConsRank-package, 2
*Topic **Kemeny**
  kemeny, 22
*Topic **Median ranking**
  ConsRank-package, 2
*Topic **Median**
  BBFULL, 7
  DECOR, 13
  FASTDECOR, 18
*Topic **Permutation**
  polyplot, 28
*Topic **Quick**
  QuickCons, 29
*Topic **Tau**
  Tau_X, 34
*Topic **Tau_X rank correlation coefficient**
  ConsRank-package, 2
*Topic **algorithms**
  DECOR, 13
*Topic **algorithm**
  FASTcons, 17

QuickCons, 29
*Topic **coefficient**
  Tau_X, 34
*Topic **correlation**
  Tau_X, 34
*Topic **datasets**
  APAFULL, 4
  APAred, 5
  BU, 9
  EMD, 16
  German, 21
  Idea, 21
  sports, 32
  USAranks, 35
*Topic **distance**
  kemeny, 22
*Topic **evolution**
  DECOR, 13
  FASTDECOR, 18
*Topic **median**
  EMCons, 15
*Topic **modal**
  tabulaterows, 33
*Topic **polytope**
  polyplot, 28
*Topic **ranking**
  BBFULL, 7
  DECOR, 13
  EMCons, 15
  FASTDECOR, 18
  tabulaterows, 33
*Topic **rank**
  Tau_X, 34

APAFULL, 4
APAred, 5

BBconsensus, 5
BBconsensus2, 6
BBFULL, 7, 16
INDEX

branches, 8
BU, 9

childclosint, 9
childtie, 10
comincost, 11
cominpmatr, 11, 32
ConsRank (ConsRank-package), 2
ConsRank-package, 2
crossover, 12

DECOR, 13
DECORcore, 14

EMCons, 7, 14, 15, 18
EMD, 16

FASTcons, 8, 14, 16, 17, 30
FASTDECOR, 18
findbranches, 20
findconsensusBB, 20

German, 21

Idea, 21

kemeny, 22, 34
kemenydesign, 23
kemenyscore, 24

labels, 25

mutaterand1, 26

Penalty, 26
PenaltyBB2, 27
polyplot, 28

QuickCons, 8, 14, 16, 18, 29, 30

reordering, 30
ReorderingBB, 31

scorematrix, 24, 31

sports, 32

tabulaterows, 12, 28, 33
Turism, 32

USAranks, 35