Package 'dst'

November 25, 2018

Type Package

Title Using Dempster-Shafer Theory

Encoding UTF-8

Version 1.3

Date 2018-12-08

Author Claude Boivin, Stat.ASSQ <webapp.cb@gmail.com>

Maintainer Claude Boivin <webapp.cb@gmail.com>

Description Using Dempster-Shafer Theory of Evidence, also called ``Theory of Belief Functions". Basic probability assignments, or mass functions, can be defined on the subsets of a set of possible values. Two mass functions can be combined using Dempster's rule of combination. A mass function can be extended to a larger frame. Marginalization, i.e. reduction to a smaller frame can also be done. These features can be combined to analyze small belief networks and take into account situations where information cannot be satisfactorily described by probability distributions.

License GPL (>= 2)

BugReports https://github.com/RAPLER/dst-1/issues

Collate 'addTobca.R'
'bca.R'
'bcaRel.R'
'belplau.R'
'decode.R'
'dotprod.R'
'doubles.R'
'dsrwon.R'
'dst.R'
'elim.R'
'encode.R'
'extmin.R'
'inters.R'
'marrayToMatrix.R'
'matrixToMarray.R'
'nameRows.R'
'nzdsr.R'
'plautrans.R'
'productSpace.R'
'reduction.R'
'shape.R'
'tabresul.R'

RoxygenNote 6.1.1

Suggests testthat,
knitr,
rmarkdown,
igraph

VignetteBuilder knitr

R topics documented:

`addTobca` *Add some elements of 0 mass to an existing mass function*

Description

Given a previously defined mass function (bca), the user may want to add some elements of the set of possible values or some subsets, even if they have zero mass value. This feature is useful, for example, to examine the plausibility results of these elements or subsets of zero mass value.

Usage

```
addTobca(x, f)
```

Arguments

x	A basic chance assignment mass function (see bca). It can also be the normalized result of the combination of two mass functions by Dempster's Rule.
f	A matrix constructed in a boolean style (0,1) or a boolean matrix. The number of columns of the matrix f must match the number of columns of the tt matrix of x (see bca). Each row of the matrix identify a subset of the set of possible values.

Value

The original bca mass function x augmented with the added subsets defined by f.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
y <- bca(f=matrix(c(1,0,0,1,1,1),nrow=2, byrow = TRUE),
m=c(0.6, 0.4),  cnames = c("a", "b", "c"), varnb=1)
addTobca(y, matrix(c(0,1,0,0,0,1, 0,1,1), nrow=3, byrow = TRUE))
x <- bca(f=matrix(c(0,1,1,1,1,0,1,1,1),nrow=3,
byrow = TRUE), m=c(0.2,0.5, 0.3),
cnames =c("a", "b", "c"), varnb=1)
xy <- dsrwon(x,y)
xy1 <- addTobca(nzdsr(xy), matrix(c(0,1,0,0,0,1), nrow=2, byrow = TRUE))
xy1
addTobca(x, f = diag(1,  ncol(x$tt) ) ) # add all singletons
```

---

bca *Basic chance assignment mass function*

Description

Function bca is used to define subsets of a finite set Θ of possible values and to assign their corresponding mass value.
The set Θ is called the frame of discernment. Each subset $A of \Theta$ with a positive mass value is called a focal element or a proposition. The associated mass value is a number of the (0,1] interval, called "basic chance assignment" (the basic probability assignment of Shafer's book). All other subsets that have not received a positive mass value are assumed to have a mass of zero value.

Usage

```
bca(f, m, cnames = NULL, infovaluenames = NULL, con = NULL,
  varnb = NULL, infovar = NULL, infovarnames = NULL,
  inforel = NULL)
```

Arguments

f	A (0,1)-matrix or a boolean matrix. The number of columns must match the number of elements (values) of the frame of discernment Θ. Each row is a subset of Θ. The last row is the frame Θ, represented by a vector of 1's.
m	A vector of masses of length equal to the number of rows of the matrix f. The values of m must lie in the interval (0,1] and must add to one. The mass m(k) represents the chance value allotted to the proposition represented by the row k of the matrix f.
cnames	A character vector containing the names of the elements of the frame of discernment Θ. The length must be equal to the number of elements of Θ. The names are searched in the infovaluenames parameter first. If NULL, column names of the matrix f are taken if present. Otherwise, names are generated.
infovaluenames	Name and value names of the variable. See bcaRel.
con	The measure of conflict. 0 by default.
varnb	The number given to the variable. 0 if omitted.
infovar	A two-column matrix containing variable identification numbers and the number of elements of the variable. Generated if omitted.
infovarnames	The name of the variable. Generated if omitted.
inforel	Not used. Defined within function bcaRel.

Value

An object of class bcaspec:

- tt The table of focal elements f. Rownames of the matrix of focal elements are generated from the column names of the elements of the frame. See nameRows for details.
- spec A two column matrix. First column: numbers given to the subsets, 1 to nrow(f). Second column: the mass values of the subsets.
- con The measure of conflict.
- infovar The number of the variable and the size of the frame of discernment.
- infovaluenames The names of the elements of the frame of discernment of the variable (the column names of the tt matrix).
- inforel Set at 0. used in function bcaRel.

Author(s)

Claude Boivin, Stat.ASSQ

References

- Shafer, G., (1976). A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey, p. 38: Basic probability assignment.
- Guan, J. W. and Bell, D. A., (1991). Evidence Theory and its Applications. Elsevier Science Publishing company inc., New York, N.Y., p. 29: Mass functions and belief functions

Examples

```
f<- t(matrix(c(1,0,1,1),ncol=2))
m<- c(.9,.1)
cnames <- c("yes","no")
bca(f, m)
bca(f, m, cnames)
bca(f, m, cnames, varnb = 1)
x <- bca(f=matrix(c(0,1,1,1,1,0,1,1,1),nrow=3,
byrow = TRUE), m=c(0.2,0.5, 0.3),
cnames =c("a", "b", "c"), varnb = 1)
y <- bca(f=matrix(c(1,0,0,1,1,1),nrow=2,
byrow = TRUE), m=c(0.6,0.4),
cnames =c("a", "b", "c"),infovarnames = "y", varnb = 1)
frame <- bca(matrix(c(1,1,1), nrow=1), m=1, cnames = c("a","b","c"))
```

---

bcaRel *Representation of a mass function in a product space*

Description

This function is used to represent a mass function which establish a relation between two or more variables in their product space.

Usage

```
bcaRel(tt, spec, infovar, infovarnames = NULL, relnb = NULL)
```

Arguments

tt	A (0,1)-matrix or a boolean matrix establishing the relation between two or more variables. The matrix is constructed by placing the variables side by side, as in a truth table representation.
spec	A two-column matrix. First column: numbers given to the subsets. Second column: the mass values of the subsets. A subset number and its associated mass value are repeated to match the number of elements of the subset.
infovar	A two column matrix containing variable identification numbers and the number of elements of each variable. The identification numbers must be ordered in increasing number.
infovarnames	The names of the variables. If omitted, variables are named v1, v2, etc.
relnb	A number given to the relation. Set at 0 if omitted.

Value

An object of class bcaspec. This is a list containing the following components:

- con The measure of conflict.
- tt The resulting table of subsets. Rownames of the matrix of subsets are generated from the column names of the elements of the product frame. See nameRows for details.
- spec The resulting two-column matrix of specification numbers with associated mass values.
- infovar The two-column matrix of variables number and size given in the input data.

- infovaluenames A list of the names of the variables with the name of the elements of their frame of discernment.
- inforel A two-column matrix containing the relation number and the depth (number of variables) of the relation.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
# A logical implication rule
# A typical relation between two variables in the context of expert systems is the
# logical implication \code{(a -> b)}. Let us suppose
# that \code{a} stands for \code{Rain: {yes, no}} and \code{b} stands for
# \code{RoadWorks: {yes, no}}. From experience,
# I am 75 % sure that there will be RoadWorks if there is no rain.
## 1. The tt table of the logical implication
 ttrwf <- matrix(c(0,1,1,0,1,0,1,0,1,0,0,1,1,1,1,1),
 nrow=4, byrow = TRUE,
 dimnames = list(NULL, c("rWdy", "rWdn", "Ry", "Rn")) )
 ## The mass distribution
 specrw <-  matrix(c(1,1,1,2,0.75,0.75,0.75,0.25), ncol = 2,
 dimnames = list(NULL, c("specnb", "mass")))
 ## Variables numbers and sizes
 inforw <- matrix(c(4,5,2,2), ncol = 2,
 dimnames = list(NULL, c("varnb", "size")) )
bcaRel(tt = ttrwf, spec = specrw, infovar = inforw,
 infovarnames = c("RdWorks", "Rain"), relnb = 6)
```

`belplau` *Calculation of the degrees of Belief and Plausibility*

Description

Degrees of Belief `Bel` and Plausibility `Pl` of the focal elements of a mass function are computed. The ratio of the plausibility of a focal element against the plausibility of its contrary is also computed. Subsets with zero mass can be excluded from the calculations.

Usage

```
belplau(x, remove = FALSE)
```

Arguments

`x`	A basic chance assignment mass function (see `bca`).
`remove`	= TRUE: Exclude subsets with zero mass.

Details

The degree of belief `Bel` is defined by:

$$bel(A) = Sum((m(B); B \subseteq A))$$

for every subset B of A.
The degree of plausibility `pl` is defined by:

$$pl(A) = Sum[(m(B); B \cap A\neg]$$

for every subset `B` of the frame of discernment.
The plausibility ratio of a focal element `A` versus its contrary `not A` is defined by: $Pl(A)/(1 - Bel(A))$.

Value

A matrix of `M` rows by 3 columns is returned, where `M` is the number of focal elements:

- Column 1: the degree of belief `Bel`;
- Column 2: the degree of Plausibility `Pl`;
- Column 3: the Plausibility ratio

Author(s)

Claude Boivin, Stat.ASSQ

References

- Shafer, G., (1976). A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey, p. 39-43.
- Williams, P., (1990). An interpretation of Shenoy and Shafer's axioms for local computation. International Journal of Approximate Reasoning 4, pp. 225-232.

Examples

```
x <- bca(f=matrix(c(0,1,1,1,1,0,1,1,1),nrow=3,
byrow = TRUE), m=c(0.2,0.5, 0.3),
cnames =c("a", "b", "c"), infovarnames = "x", varnb = 1)
belplau(x)
y <- bca(f=matrix(c(1,0,0,1,1,1),nrow=2,
byrow = TRUE), m=c(0.6, 0.4),
cnames = c("a", "b", "c"),  infovarnames = "y", varnb = 1)
belplau(nzdsr(dsrwon(x,y)))
print("compare all elementary events")
xy1 <- addTobca(nzdsr(dsrwon(x,y)),
matrix(c(0,1,0,0,0,1), nrow=2, byrow = TRUE))
belplau(xy1)
```

decode *Find the value in base 10 of a number coded in another base*

Description

The aplDecode function of the project APL in R (https://rpubs.com/deleeuw/158476) has been adapted to follow the standard implementation of the APL decode function.

Usage

```
decode(base, ind)
```

Arguments

base	A scalar or a numeric vector which describes the number system in which the data is coded.
ind	The value to decode represented by a numeric vector in the base system.

Details

If the base value is a number system, namely base 2, we need only to enter a scalar, which is treated to match the length of the expression to decode. If the base value is a number system, e.g. base 2, we need only to enter it as a scalar, which is then processed to match the length of the expression to decode. If length(ind) is less than length(base), zeroes are added to the left of the vector ind to match the length of the two vectors. And vice-versa.

Value

A scalar representing the conversion of the coded number ind to its decimal representation.

Author(s)

Claude Boivin, Stat.ASSQ.

References

- Jan de Leeuw and Masanao Yajima (March 07, 2016) *APL in R (Version 009)*, Source code. https://rpubs.com/deleeuw/158476
- L. Gilman and A. J. Rose.(1974): *APL an Interactive Approach*, Second Edition, John Wiley, New York.
- APL 68000 Level II language manual. MicroAPL Ltd. 1990.

Examples

```
decode(c(2,2,2,2), c(1,0,1,1)) #   Find the base 10 value of the base 2 number 1011.
decode(2, c(1,0,1,1))  # left argument is extended to vector c(2,2,2,2)
decode(c(365,24,60), c(2,1,57)) # transform 2 days 1 h 57 min in minutes
decode(c(365,24,60), c(1,57))   # right vector extended
decode(c(24,60), c(2,1,57))     # left vector extended
decode(1.5, c(1,2,3)) # polynomial 1*x^2 +2*x +3 evaluated at x=1.5
```

dotprod *Generalized inner product of two matrices*

Description

The generalized inner product of two matrices combines two operators in the same manner as the classical inner product defined for the multiplication of two matrices. The number of rows of the second matrix must be equal the number of columns of the first matrix.

Usage

```
dotprod(x, y, g, f)
```

Arguments

x	A matrix of M rows by K columns.
y	A matrix of K rows by N columns.
g	Any operator: +, -, *, /, &, |, ==, <=, paste etc.
f	Any operator: +, -, *, /, &, |, ==, <=, paste etc.

Value

The result of the generalized inner product is returned.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
print("Standard matrix product")
x <- y <- matrix(c(1:6), nrow = 2, byrow = TRUE)
dotprod(x, t(y), g = "+", f = "*")  ## same as x %*% t(y)
print("Find some data x2 in the rows of a larger matrix y2")
x2 <- matrix(c(1,0,0,1,1,1), nrow = 2, byrow = TRUE)
y2 <- matrix(c(1,0,0,0,1,0,1,1,0,0,1,1,1,1,1),
nrow = 5, byrow = TRUE)
(1:nrow(y2)) * dotprod(x2, t(y2), g = "&", f = "==")

print("Find some names in a long list")
team_names <- matrix(c("Patrick", "Dole", "Amanda",
 "Dole", "Robert", "Calvin", "Alvina", "Klein",
  "Robert", "Gariepy", "Nellie", "Arcand"),
   ncol = 2, byrow = TRUE)
colnames(team_names) <- c("First_name", "Last_name")
print("Where in the list are the person with first name Robert and where are the Doles?")
BobandDoles <- matrix(c("Robert", "", "", "Dole"),
 ncol = 2, byrow = TRUE)
dotprod(team_names, t(BobandDoles),g="|",f="==") * (1:nrow(team_names))
```

doubles *Remove duplicate rows in a two-dimensional table*

Description

Remove duplicate rows in a two-dimensional table

Usage

```
doubles(x)
```

Arguments

x	A matrix of numeric, character or logical type.

Value

The submitted matrix with duplicated rows removed from.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
td0<-matrix(c(rep(c(1,0,1),times=3),0,0,1,1,1,1, 1,1,1),ncol=3,byrow=TRUE)
(doubles(td0))
td1<-matrix(c(rep(c(1,0,1),times=3),0,0,1,1,1,1),ncol=3,byrow=TRUE)
(doubles(td1))
td2<-matrix(c(1:3, 1:3,4:6,1:3),nrow=4,byrow=TRUE)
(doubles(td2))
td3<-matrix(c("d","e","f", rep(c("a","b","cc"),times=3),"g","h","i"),nrow=5,byrow=TRUE)
(doubles(td3))
td4<-matrix(as.logical(td1),nrow=5,byrow=TRUE)
(doubles(td4))
```

dsrwon *Combination of two mass functions*

Description

The unnormalized Dempster's rule is used to combine two mass functions `mx` and `my` defined on the same frame of discernment and represented by their respective basic chance assignments `x` and `y`. Dempster's rule of combination is applied. The normalization is not done, leaving the choice to the user to normalize the results or not (for the normalization operation, see `nzdsr`).

Usage

```
dsrwon(x, y, relnb = NULL)
```

Arguments

x	A bca mass function (see bca).
y	A bca mass function (see bca).
relnb	Identification number of the relation. Can be omitted.

Details

The two bca's x and y must be defined on the same frame of discernment for the combination to take place. The relation number of the x input is given to the output result.

Value

A list of class bcaspec with these two components added:

- I12 Intersection table of subsets.
- Sort_order Sort order of subsets.

Author(s)

Claude Boivin, Stat.ASSQ

References

Shafer, G., (1976). A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey, pp. 57-61: Dempster's rule of combination.

Examples

```
x1 <- bca(f=matrix(c(0,1,1,1,1,0,1,1,1),nrow=3,
byrow = TRUE), m=c(0.2,0.5, 0.3),
cnames =c("a", "b", "c"),
infovarnames = "x", varnb=1)
x2 <- bca(f=matrix(c(1,0,0,1,1,1),nrow=2,
byrow = TRUE), m=c(0.6, 0.4),
cnames = c("a", "b", "c"),
infovarnames = "x", varnb = 1)
dsrwon(x1,x2)
```

---

dst *Manipulation and combination of belief functions*

Description

dst allows you to make basic probability assignments on subsets of a set of possibilities (events) and combine these events with Dempster's rule of combination.

Details

The main operations that can be done are:

- definition of a basic chance assignment (bca) distribution on a variable
- combination of two bca's defined on the same variable
- definition of a bca which establish a relation between two or more variables
- extension of a bca
- marginalization of a bca

Author(s)

Claude Boivin, Stat.ASSQ <webapp.cb@gmail.com>

`elim` *Reduction of a relation*

Description

This function works on a relation defined on a product of two variables or more. Having fixed a variable to eliminate from the relation, the reduced product space is determined and the corresponding reduced bca is computed.This operation is also called "marginalization".

Usage

```
elim(rel, xnb)
```

Arguments

`rel`	The relation to reduce, an object of class bcaspec.
`xnb`	Identification number of the variable to eliminate.

Value

r The reduced relation

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
wr_tt <- matrix(c(0,1,rep(0,5),rep(c(1,0),2),1,1,0,1,0,
rep(1,3),0,1,0,rep(1,6)), ncol=4, byrow = TRUE)
colnames(wr_tt) <- c("rWdy Ry", "rWdy Rn", "rWdn Ry", "rWdn Rn")
 wr_spec = matrix(c(1:7, 0.0476, 0.7619, 0.1905, 0,0,0,0),
 ncol = 2, dimnames = list(NULL, c("specnb", "mass")))
 wr_infovar = matrix(c(4,5,2,2), ncol = 2,
 dimnames = list(NULL, c("varnb", "size")) )
 wr_rel <- list(tt=wr_tt, con=0.16, spec=wr_spec,
  infovar=wr_infovar,
```

```
 infovaluenames= list(Rain=c("Ry", "Rn"), RdWorks=c("rWdy", "rWdn") ))
class(wr_rel)="bcaspec"
elim(wr_rel, xnb = 5)
elim(wr_rel, xnb = 4)

mrt_tt <- matrix(c(1,0,1,0,0,1,1,0,0,1,0,1,1,0,0,1,0,1,1,0,0,1,0,1,rep(1,4)),
ncol=4, byrow = TRUE)
colnames(mrt_tt) <- c("t6", "f6", "t8", "f8")
mrt_spec = matrix(c(1,1,1,2,2,2,3, 0.1, 0.1, 0.1, 0.7,0.7,0.7,0.2),
ncol = 2, dimnames = list(NULL, c("specnb", "mass")))
mrt_infovar =matrix(c(6,8,2,2), ncol = 2,
dimnames = list(NULL, c("varnb", "size")) )
mrt_rel <- bcaRel(tt=mrt_tt, spec=mrt_spec,
infovar=mrt_infovar,
infovarnames= c("Maintenance", "Repair") )
elim(mrt_rel, xnb = 6)
elim(mrt_rel, xnb = 8)
```

---

encode *Convert a value to its representation in another chosen base*

Description

The aplEncode function of the project APL in R (https://rpubs.com/deleeuw/158476) has been adapted to follow the standard implementation of the APL encode function.

Usage

```
encode(base, ind)
```

Arguments

base	A numeric vector which describes the number system in which we want to re-code the data.
ind	The value to convert represented by a number or a numeric vector.

Value

A vector or a matrix of the data converted.

Author(s)

Claude Boivin, Stat.ASSQ.

References

- Jan de Leeuw and Masanao Yajima (March 07, 2016) *APL in R (Version 009)*, Source code. https://rpubs.com/deleeuw/158476
- L. Gilman and A. J. Rose.(1974): *APL an Interactive Approach*, Second Edition, John Wiley, New York.
- APL 68000 Level II language manual. MicroAPL Ltd. 1990.

Examples

```
encode(c(2,2,2,2), 11)  # find the base 2 representation of number 11
encode(c(365,24,60), 2997) # convert 2997 minutes to days-hrs-min.
```

`extmin` *Extension of a relation*

Description

This function works on a mass function defined on a single variable or a relation defined onto a group of two variables or more. An extension of their space is made to a larger product space of a relation of reference. The mass function or relation to extend and the relation of reference must have at least one common variable for the extension to be made possible.

Usage

```
extmin(rel1, relRef)
```

Arguments

`rel1`	An object of class bcaspec, i.e. a mass function of one variable or a relation.
`relRef`	The relation of reference. It can be an existing relation, or it can be constructed as a vacuous function.

Details

The `relRef` parameter is used to extract all the information on the variables, namely their identification numbers and the number of elements of each variable, variables names and columns names of the tt matrix. The relation of reference `relRef` may simply be an empty relation defined on the set of variables of interest or a relation already defined.

Value

R the resulting extended relation.

Author(s)

Claude Boivin, Stat.ASSQ

References

G. Shafer and P. P. Shenoy. Local Computations in Hypertrees. School of Business, University of Kansas, Lawrence, KS, 1991. See p. 78, vacuous extension of a belief function.

Examples

```
# Naking an empty reference relation with mass(frame) = 1 and
# extending a bca to its space.
init_tt= matrix(rep(1,10),nrow=1,
dimnames =list(NULL, c("3", "2", "1", "0",
 "true", "false",  "foul", "fair",  "true", "false")) )
 init_spec <- matrix(c(1,1), ncol = 2,
 dimnames = list(NULL, c("specnb", "mass")))
 init_info <- matrix(c(3,4,7,8,4,2,2,2), ncol = 2,
  dimnames = list(NULL, c("varnb", "size")) )
 relRef <- bcaRel(tt = init_tt, spec = init_spec,
  infovar = init_info,
  infovarnames = c("Sail", "Loading", "Weather", "Repairs"),
  relnb = 0)
 # a bcaspec defined on one variable
 l_rel <- bca(f=matrix(c(1,0,1,0,1,1), ncol=2),
 m=c(0.3,0.5,0.2), cnames=c("true", "false"),
 infovar=matrix(c(4,2), ncol = 2,
 dimnames = list(NULL, c("varnb", "size"))),
 infovarnames= c("Loading"),
 inforel= matrix(c(7,1), ncol = 2,
 dimnames = list(NULL, c("relnb", "depth"))))
 z <- extmin(l_rel, relRef)
 prmatrix(t(z$tt), collab = rep("", nrow(z$tt)))
```

---

inters *Intersection of two tables of propositions*

Description

Function `inters` returns a table of the intersection between two (0,1) or boolean matrices or two vectors. The two matrices must have the same number of columns. The two vectors must be of the same length. This function generalizes the intersection of two subsets represented by boolean vectors to the intersection of two matrices of subsets.

Usage

```
inters(x, y)
```

Arguments

x	A (0,1)-matrix or a boolean matrix of M rows by K columns, or a vector of length K.
y	A (0,1)-matrix or a boolean matrix of N rows by K columns or a vector of length K.

Value

The result is a (0,1)-table of dimensions (M x K) x N). In the case of vectors, the result is a (0,1)-table of dimensions (1 x K) x 1)

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
mx<-matrix(c(0,1,0,0,1,1,1,1,1),nrow=3, byrow = TRUE, dimnames = list(NULL, c("a", "b", "c")))
 rownames(mx) <- nameRows(mx)
my<-matrix(c(0,0,1,1,1,1),nrow=2, byrow = TRUE, dimnames = list(NULL, c("a", "b", "c")))
 rownames(my) <- nameRows(my)
inters(mx,my)
b1 <- c(FALSE, TRUE, TRUE)
b2 <- c(TRUE, TRUE, FALSE)
names(b1) <- names(b2) <- c("c1","c2","c3")
inters(b1,b2)
x3<-matrix(c(1,1,0,1), ncol=2, dimnames=list(NULL, c("a","b")))
y3<-matrix(c(0,1,1,1), ncol=2, dimnames=list(NULL, c("a","b")))
inters(x3,y3)
x4 <-matrix(c(1,0,1,1,1,1,1,1),nrow=2, byrow = TRUE, dimnames = list(NULL, c("a", "b", "c","d")))
y4 <-matrix(c(1,0,0,1,1,1,1,1),nrow=2, byrow = TRUE, dimnames = list(NULL, c("a", "b", "c","d")))
inters(x4,y4)
```

---

marrayToMatrix *Transformation of an array data to a matrix-represented relation*

Description

The array representation or product space representation is converted to the matrix representation of the corresponding relation.

Usage

```
marrayToMatrix(mtt, infovar)
```

Arguments

mtt	The matrix tt of the relation in multiarray format
infovar	specification of the number and size of each variable

Value

tt The matrix representation of the data.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
 wr_infovar = matrix(c(4,5,2,2), ncol = 2,
 dimnames = list(NULL, c("varnb", "size")) )
 mtt <- array(c(0,1,0,0,0,0,0,1,0,1,0,1,1,0,1,0,1,1,1,0,1,0,1,1,1,1,1,1), c(2,2,7),
 dimnames = list( RdWorks=c("rWdy", "rWdn") , Rain=c("Ry", "Rn"), ev=1:7))
z <- marrayToMatrix(mtt, wr_infovar)
```

matrixToMarray *Transformation of the tt matrix of a relation*

Description

The matrix representation of a relation is converted to the array representation or product space representation.

Usage

```
matrixToMarray(rel)
```

Arguments

rel	An object of class bcaspec, i.e. a mass function of one variable or a relation.

Value

mtt The array (product space) representation of the tt matrix.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
wr_tt <- matrix(c(0,1,rep(0,5),rep(c(1,0),2),1,1,0,1,0,
rep(1,3),0,1,0,rep(1,6)), ncol=4, byrow = TRUE)
colnames(wr_tt) <- c("rWdy Ry", "rWdy Rn", "rWdn Ry", "rWdn Rn")
 wr_spec = matrix(c(1:7, 0.0476, 0.7619, 0.1905, 0,0,0,0),
 ncol = 2, dimnames = list(NULL, c("specnb", "mass")))
 wr_infovar = matrix(c(4,5,2,2), ncol = 2,
 dimnames = list(NULL, c("varnb", "size")) )
 wr_rel <- list(tt=wr_tt, con=0.16, spec=wr_spec,
  infovar=wr_infovar,
  infovaluenames= list( RdWorks=c("rWdy", "rWdn") , Rain=c("Ry", "Rn")))
 class(wr_rel)="bcaspec"
z <- matrixToMarray(wr_rel)
```

nameRows *Using the column names of a matrix to construct names for the rows*

Description

This function determines the name of a row from all the columns that have a 1 for that row.

Usage

```
nameRows(f)
```

Arguments

f	A (0,1)-matrix or a boolean matrix.

Details

The row containing all one's is called "frame", to avoid too long a character string. The empty set is named by its code "u00f8". The "+" sign is used to represent the logical "or" operator. The space " " is used to represent the logical "and" operator. Note that in the case of a product space defined on many variables, row labels will be very long.

Value

The result is a character vector of the labels proposed for the rows of the matrix `f`. The length of the result is `nrow(f)`.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
f <- matrix(c(0,0,0,1,0,0,0,0,1,1,0,1,1,1,1),ncol=3, byrow = TRUE)
colnames(f) <- c("A","B","C")
rownames(f) <-nameRows(f)
f
f2 <- matrix(c(0,0,0,1,0,0,0,0,1,1,0,1),ncol=3, byrow = TRUE)
colnames(f2) <- c("A2","B2","C2")
rownames(f2) <-nameRows(f2)
f2
```

---

nzdsr *Normalization of a bca mass function*

Description

It may occur that the result of the combination of two mass functions with Dempater'Rule of combination contains a non-zero mass allocated to the empty set. The function `nzdsr` normalizes the result of function `dsrwon` by dividing the mass value of the non-empty subsets by 1 minus the mass of the empty set.

Usage

```
nzdsr(x)
```

Arguments

x	A mass function, i.e. a list of class bcaspec..

Value

The normalized bca mass function.

Author(s)

Claude Boivin, Stat.ASSQ

References

Shafer, G., (1976). A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey, pp. 57-61: Dempster's rule of combination.

Examples

```
x1 <- bca(f=matrix(c(1,0,1,1),nrow=2, byrow = TRUE),
m=c(0.9,0.1), cnames =c("yes", "no"),
infovarnames = "x", varnb = 1)
x2 <- bca(f=matrix(c(0,1,1,1),nrow=2, byrow = TRUE),
m=c(0.5,0.5), cnames =c("yes", "no"),
infovarnames = "x", varnb = 1)
print("combination of x1 and x2")
x1x2 <- dsrwon(x1,x2)
nzdsr(x1x2)

print("normalization of a bca definition.")
y2 <- bca(f=matrix(c(0,0,0,1,0,0,1,1,1),nrow=3,
byrow = TRUE), m=c(0.2,0.5,0.3),
cnames =c("a", "b", "c"), varnb = 1)
nzdsr(y2)
```

---

plautrans *Plausibility transformation of the singletons of a frame*

Description

Given a mass function defined on some subsets of a frame Θ, the application of the plausibility transformation to the singletons of Θ yields the probability distribution associated with the this mass function.

Usage

```
plautrans(x)
```

Arguments

x	A bca mass function.

Details

We compute the plausibility measure of all the singletons of the frame of discernment. The probability distribution of the singletons is obtained from their plausibility measures.

Value

The matrix of singletons with their plausibility transformation added in the last column.

Author(s)

Claude Boivin, Stat.ASSQ

References

Cobb, B. R. and Shenoy, P.P. (2006). On the plausibility transformation method for translating belief function models to probability models. Journal of Approximate Reasoning, 41(3), April 2006, 314–330.

Examples

```
x <- bca(f=matrix(c(0,1,1,1,1,0,1,1,1),nrow=3,
byrow = TRUE), m=c(0.2,0.5, 0.3),
cnames =c("a", "b", "c"),
infovarnames = "x", varnb = 1)
plautrans(x)
```

---

productSpace *Product space representation of a relation*

Description

This utility function takes the input matrix of a relation between two or more variables and yields its product space representation.

Usage

```
productSpace(tt, specnb, infovar)
```

Arguments

tt	A (0,1) or boolean matrix, where the values of the variables put in relation are set side by side, as in a truth table.
specnb	A vector of integers ranging from 1 to k, where k is the number of subsets of the tt matrix. Values must start at one and can be increased only by 0 or 1. They determine the partitioning of the rows of the tt matrix between the k subsets.
infovar	A two-column matrix containing identification numbers of the variables and the number of elements of each variable (size of the frame).

Value

The matrix of the product space representation of the relation.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
 ttfw= matrix(c(1,0,1,0,0,1,0,1,1,1,1,1),nrow=3,
  byrow = TRUE,
  dimnames =list(NULL, c("foul", "fair", "foul", "fair")) )
 specfw = c(1,1,2)
 infovarfw =matrix(c(5,7,2,2), ncol = 2,
 dimnames = list(NULL, c("varnb", "size")) )
productSpace(tt=ttfw, specnb=specfw, infovar=infovarfw)
```

---

reduction *Summary of a vector for any operator.*

Description

This utility function is used to obtain a summary of a vector of data for many operators. The function is taken from the project APL in R (https://rpubs.com/deleeuw/158476).

Usage

```
reduction(x, f = "+")
```

Arguments

x	A vector of numbers or character strings.
f	The operator. Must be compatible with the type of input vector (numeric or character)

Value

The result of applying the chosen operator to all the elements of the vector is an object of length 1.

Author(s)

Claude Boivin, Stat.ASSQ.

References

- Jan de Leeuw and Masanao Yajima (March 07, 2016) *APL in R (Version 009)*, Source code. https://rpubs.com/deleeuw/158476
- G. Helzer. (1989): *An Encyclopedia of APL*, second edition, I-APL LTD, St. Albans, G.B.
- L. Gilman and A. J. Rose.(1974): *APL an Interactive Approach*, Second Edition, John Wiley, New-York.

Examples

```
reduction(c(1,2,3,4), f="-")
reduction(c(1,0,1,1,0), f="|")
reduction(c("a", "b", "c"), f="paste")
```

shape *Obtain dimensions of an array or length of a vector with a single command*

Description

shape returns sizes of each dimension of given array or the length of a given vector. The function is taken from the project APL in R (https://rpubs.com/deleeuw/158476).

Usage

```
shape(a)
```

Arguments

a	An array or a vector.

Value

The dimensions of the array a or the length of the vector a.

Author(s)

Claude Boivin, Stat.ASSQ.

References

- Jan de Leeuw and Masanao Yajima (March 07, 2016) *APL in R (Version 009)*, Source code. https://rpubs.com/deleeuw/158476
- G. Helzer. (1989): *An Encyclopedia of APL*, second edition, I-APL LTD, St. Albans, G.B.
- L. Gilman and A. J. Rose.(1974): *APL an Interactive Approach*, Second Edition, John Wiley, New-York.

Examples

```
shape(array(c(1:6), c(2,3)))
shape(c("a", "b"))
```

tabresul *Prepare a table of results*

Description

This utility function is a more detailed version of the belplau function. Different tables of measures of belief, plausibility and of the plausibility ratio can be obtained, namely by removing subsets with zero mass if present, or by asking for singletons only.

Usage

```
tabresul(x, singletonsOnly = FALSE, removeZeroes = FALSE)
```

Arguments

x	A bca mass function
singletonsOnly	= TRUE reduces the table of results to elementary events (singletons).
removeZeroes	= TRUE removes subsets with 0 mass.

Value

A list of two elements:

- mbp: The table of focal elements with the addition of their associated mass, degree of belief, plausibility and the plausibility ratio.
- con The measure of conflict between subsets.

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
x <- bca(f=matrix(c(0,1,1,1,1,0,1,1,1),nrow=3,
byrow = TRUE), m=c(0.2,0.5, 0.3),
cnames =c("a", "b", "c"),
infovarnames = "x", varnb = 1)
y <- bca(f=matrix(c(1,0,0,1,1,1),nrow=2,
byrow = TRUE), m=c(0.6, 0.4),
cnames = c("a", "b", "c"), infovarnames = "y", varnb = 1)
xy <- dsrwon(x,y)
xyNorm <- nzdsr(xy)
tabresul(xyNorm)
## print("Show all elementary events")
xy1 <- addTobca(nzdsr(dsrwon(x,y)),
matrix(c(0,1,0,0,0,1),
nrow=2, byrow = TRUE))
tabresul(xy1)
## print("Remove focal elements with 0 mass")
tabresul(xy1, removeZeroes = TRUE)
print("Retain singletons only")
tabresul(xy1, singletonsOnly = TRUE)
```

Index