Package 'dst'

April 1, 2018

Type Package

Title Using Dempster-Shafer Theory

Version 0.4

Date 2018-xx-xx

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

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

Description This package 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.

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'
'elim.R'
'encode.R'
'extmin.R'
'inters.R'
'nameRows.R'
'nzdsr.R'
'plautrans.R'
'productSpace.R'
'reduction.R'
'shape.R'
'tabresul.R'

RoxygenNote 6.0.1

Suggests testthat,
knitr,
rmarkdown,
igraph

VignetteBuilder knitr

R topics documented:

addTobca 2
bca 3
bcaRel 4
belplau 6
decode 7
dotprod 8
doubles 9
dsrwon 10
elim 11
encode 12
extmin 13
inters 14
nameRows 15
nzdsr 16
plautrans 17
productSpace 18
reduction 19
shape 20
tabresul 21

Index **22**

`addTobca` *Add focal elements with 0 mass*

Description

This utility function allows to expand a bca definition with focal elements of zero mass. This is useful when we want to compare the plausibility results of some focal elements which do not already appear because their belief mass is 0.

Usage

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

Arguments

`x`	A belief function in its bca form (see `bca`). It can be a bca definition or the normalized result of the combination of bca's by Dempster'sRule.
`f`	A matrix constructed in a boolean style (0,1) or a boolean matrix. The number of columns of the matrix must match the number of elements (values) of the frame of discernment Θ of `x`. @return The original bca `x` augmented with the added focal elements 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 distribution*

Description

This function assigns their corresponding mass to some subsets of a finite set Θ of possible values. The set Θ is called the frame of discernement. Each subset $A of \Theta$ is called a focal element or a proposition. The associated mass is a number in the (0,1] interval, called here "basic chance assignment" (the basic probability assignment of Shafer's book).

Usage

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

Arguments

`f`	A matrix constructed in a boolean style (0,1) or a boolean matrix. The number of columns of the matrix must match the number of elements (values) of the frame of discernment Θ. Each row of the matrix is a focal element, i.e. a subset of the frame of discernment described by a vector of (0,1). The last line 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 (0,1] interval. The sum of the elements of `m` must be 1. The mass `m[k]` represents the chances allowed to the proposition represented by the row k of the matrix `f`.
`cnames`	A character vector of names of the elements of the frame of discernment, of length equal to the number of elements of the frame Θ. If NULL, takes column names of the matrif f if present. Otherwise, names are generated.
`con`	The measure of conflict. Set at 0 by default.
`varnb`	A variable number. The variable number will be used when combining bca's defined on two or more frames of discernment. See `productSpace`. Set at value 0 if not given.
`infovar`	A two column matrix containing variable identification numbers and the number of elements of the variable.
`infovarnames`	A name given to the variable. Named "v1" if omitted.
`infovaluenames`	Not used. Defined within function `bcaRel`.
`inforel`	Not used. Defined within function `bcaRel`.

Value

The result is the representation of a belief function by its basic chance assignment of propositions. It is an object of class `bcaspec`, a list of six elements:

- $tt The table of focal elements f. Rownames of the matrix of focal elements are created from the column names of the elements of the frame. See `nameRows` for details.
- $spec A two column matrix. First column contains specification numbers: 1 to `nrow(f)`. Second column contains the mass vector.
- $infovar The variable number and the size of the frame of discernment.
- $infovaluenames The names of the elements of the frame of the variable (the column names of the tt matrix).
- $inforel Set at 0. used in function `bcaRel`

@details The Basic chance assignment distribution of a variable can also be obtained using the more general 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.

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 belief function in a product space by its basic chance assignment distribution*

Description

A relation between two or more variables can be established in their product space and a mass function defined accordingly.

Usage

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

Arguments

`tt`	A (0,1) or logical matrix establishing the relation between two or more variables. The relation is described by a matrix input table of (0,1), where the variables are put side by side, as in a truth table representation.
`spec`	A two column matrix. First column contains subsets numbers. Second column contains the mass value associated with each subset of the relation. 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 elements:

- $con The measure of conflict. Set at 0 by default.
- $tt The resulting table of focal elements. Rownames of the matrix of focal elements are created from the column names of the elements of the product frame.
- $spec The resulting two column matrix of specification numbers with associated masses.
- $infovar The two column matrix given in the input data.
- infovaluenames A list of the names of the variables with the value name of each element.
- $inforel A two column matrix containing variable numbers 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 is the
# logical implication rule (a -> b). let us suppose
# that a stands for Rain: {yes, no} and b stands for
# oadWorks: {yes, no}. From experience in my region,
# I am 75 % sure that there will be RoadWorks if no Rain.
## The truth table
 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 belief function are computed. Then the ratio of the plausibility of a focal element against the plausibility of its contrary is computed. Focal elements with zero mass can be excluded from the calculations.

Usage

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

Arguments

x	A belief function in its bca form (see bca).
remove	= TRUE: Focal elements with 0 mass are excluded from the calculations.

Details

The degree Belief Bel is defined by:
$bel(A) = Sum(m(B); B <= A)$, for every subset A of the frame of discernment. The plausibility function pl is defined by:
$pl(A) = Sum(m(B); BandAnotempty$, for every subset A of the frame of discernment. The plausibility ratio of a focal element A versus its contrary ~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 number system*

Description

This utility function is used to find the value in base 10 of a number represented in another number system. This code has been adapted from the `aplDecode` R function of Jan de Leeuw. It follows the standard decode implementation of the APL language.

Usage

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

Arguments

`base`	A scalar or a numeric vector which describe 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 `length(ind)` is less than `length(base)`, `0` values 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.
- Author of the aplDecode function: Jan de Leeuw `http://www.codecollector.net/view/8A8D9395-0F66-4706-A23E-C588151E8423-95744-0000429BCF33A153`.

References

- 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.

See Also

Jan de Leeuw and Masanao Yajima: https://rpubs.com/deleeuw/158476.

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 duplicates rows in a two dimensional table*

Description

The submitted tables can be matrices of numeric character or logical types.

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 belief functions*

Description

The unnormalized Dempster's rule is used to combine two belief functions Bel1 and Bel2 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 (See `nzdsr`).

Usage

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

Arguments

`x`	A belief function in its bca form (see `bca`).
`y`	A belief function in its bca form.
`infovarnames`	A name can be given to the resulting variable. Named "v1" if missing.
`relnb`	A number can be given to the resulting relation. if omitted, the numbers of the input relations will be kept.

Details

The two bca's `a` and `b` must be defined on the same frame of discernment for the combination to take place.

Value

A list of five elements:

- $I12 A table of intersections between subsets.
- $sort_order Indices for the sort of the propositions.
- $con: the measure of conflict between beliefs.
- $inforel. A two column matrix containing variable numbers and the depth of the relation.

@details The relation number of the x input is given to the result.

Author(s)

Claude Boivin, Stat.ASSQ

References

Shafer, G., (1976). A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey, p. 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)
```

`elim` *Reduction of a relation*

Description

This function starts with a relation defined on a product space. Having chosen a variable to eliminate, the reduced product space is determined and the remaining bca is calculated.

Usage

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

Arguments

`rel`	the relation to marginalize, an object of class bcaspec.
`xnb`	Identification by its number of variable to eliminate.

Value

The reduced relation

Author(s)

Claude Boivin, Stat.ASSQ

Examples

```
rel_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(rel_tt) <- c("rWdy Ry", "rWdy Rn", "rWdn Ry", "rWdn Rn")
 rel_spec = matrix(c(1:7, 0.0476, 0.7619, 0.1905, 0,0,0,0),
 ncol = 2, dimnames = list(NULL, c("specnb", "mass")))
 rel_infovar = matrix(c(4,5,2,2), ncol = 2,
 dimnames = list(NULL, c("varnb", "size")) )
 rel_rel <- list(tt=rel_tt, con=0.16, spec=rel_spec,
  infovar=rel_infovar,
  infovaluenames= list(Rain=c("Ry", "Rn"), RdWorks=c("rWdy", "rWdn") ))
 class(rel_rel)="bcaspec"
 elim(rel_rel, xnb = 5)
 elim(rel_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 a representation in another number system*

Description

This utility function is used to convert data in another number system. This code has been adapted from the aplEncode R function of Jan de Leeuw. It follows the standard encode implementation of the APL language.

Usage

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

Arguments

base	A numeric vector which describe the number system in which we want to recode the data.
ind	The value to convert represented by a number or a numeric vector.

Value

A vector or a matrix representing the conversion of the data.

Author(s)

- Claude Boivin, Stat.ASSQ.
- Author of the aplEncode function: Jan de Leeuw http://www.codecollector.net/view/8A8D9395-0F66-4706-A23E-C588151E8423-95744-0000429BCF33A153.

References

- 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.

See Also

Jan de Leeuw and Masanao Yajima: https://rpubs.com/deleeuw/158476.

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 relation rel1 defined onto a group of one or more variables to extend it to the product space of another group of variables. This other group of variables must contain at least one of the variables of rel1 for the extension to be made possible.

Usage

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

Arguments

`rel1`	A relation, an object of class bcaspec.
`relRef`	The relation of reference used to extract the varables names and columns names of the tt matrix.

Details

The relation of reference relRef may simply be an empty relation defined on the set of variables of interest or a relation already defined. The relRef parameter normally contains all the information on the variables, namely their identification numbers and the number of elements of each variable ($infovar parameter). The relRef relation also contains the names of the variables and of the columns of the tt matrix.

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, vacuus extension of a belief function.

Examples

```
# making an empty reference relation with mass(frame) = 1
init_tt= matrix(rep(1,10),nrow=1,
dimnames =list(NULL, c("0", "1", "2", "3",
"true", "false", "foul", "fair", "true", "false")) )
 init_spec <- matrix(c(1,1), ncol = 2,
 dimnames = list(NULL, c("specnb", "mass")))
 init_info <- matrix(c(2,4,5,6,4,2,2,2), ncol = 2,
  dimnames = list(NULL, c("varnb", "size")) )
 relRef <- bcaRel(tt = init_tt, spec = init_spec,
  infovar = init_info,
  infovarnames = c("Delay", "Loading", "Forecast", "Maintenance"),
  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"))))
 extmin(l_rel, relRef)
```

`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 vector must have the same length. This function generalizes the intersection of two subsets represented by boolean vectors to two matrices of subsets.

Usage

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

Arguments

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

`nameRows` *Naming the rows of a matrix of focal elements*

Description

This function uses the column names of a matrix `f` of focal elements to construct the names of the rows.

Usage

```
nameRows(f)
```

Arguments

`f`	A boolean matrix or a matrix constructed in a boolean style (0,1). The names of the columns are the elements of a frame of discernment Θ.

Value

The result is a character vector of the names of the focal elements of the matrix `f`. The length of the result is `nrows(f)`. The set of all elements is called "frame", to avoid to long a string. The empty set is named "u00f8". The "+" sign is used instead of the "|" to represent the logical "or" operation.

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 distribution*

Description

It may occur that the resulting distribution of the combination of two bca distributions contains a non-zero mass allocated to the empty set. The function nzdsr normalizes this distribution by dividing the focal elements (other than the empty set) by 1 minus the mass of the empty set.

Usage

```
nzdsr(x, infovarnames = NULL)
```

Arguments

x	A list of class bcaspec, normally the result of the combination of two bca mass distributions that we want to normalize (see dsrwon), or simply a belief function in its bca form (see bca).
infovarnames	A name can be given to the resulting variable. Named "nv1" if missing.

Value

A list in the bca form (see bca), namely:

- $con The measure of conflict.

Author(s)

Claude Boivin, Stat.ASSQ

References

Shafer, G., (1976). A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey, p. 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 applied on the distribution of singletons*

Description

The plausibility transformation is used to obtain the probability distribution associated with a belief function.

Usage

```
plautrans(x)
```

Arguments

x	A belief function in its bca form or the normalized result of the combination of two or more belief functions (see nzdsr.

Details

First, we compute the belief and plausibility measures on all the singletons of the frame of discernment. The probability distribution of the singletons is derived from the plausibility measures of the singletons.

Value

The matrix of singletons with the 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 *Transformation of multiple side inputs to a product space definition*

Description

This utility function is used to obtain a product space definition of two or more variables initially represented by a truth table format.

Usage

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

Arguments

tt	The table of the variables being put in relation. This relation is described by a matrix input table of (0,1), where the (0,1) values of all variables are put side by side, as in a truth table.
specnb	A vector of specification numbers. Values of variable specnb start at one and are increased by 0 or 1 only. They determine the partitioning of the rows of the tt matrix.
infovar	A two-column matrix containing identification numbers of the variables and the number of elements of each variable.

Value

The thruth table matrix of the product space definition.

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. This code has been adapted from the `aplRDV` R function of Jan de Leeuw.

Usage

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

Arguments

`x`	A vector of numbers or strings.
`f`	The operator. Must be compatible with the type of 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.
- Original author of the aptRDV function: Jan de Leeuw `http://www.codecollector.net/view/8A8D9395-0F66-4706-A23E-C588151E8423-95744-0000429BCF33A153`.

References

- 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.

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

See Also

Jan de Leeuw and Masanao Yajima: `https://rpubs.com/deleeuw/158476`.

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

Description

shape returns sizes of each dimensions of given array or the length of a given vector.

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)

Original author of the aplShape function: Jan de Leeuw http://www.codecollector.net/view/8A8D9395-0F66-4706-A23E-C588151E8423-95744-0000429BCF33A153.

References

- 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.

See Also

Jan de Leeuw and Masanao Yajima: https://rpubs.com/deleeuw/158476.

Examples

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

`tabresul` *Prepare a table of results*

Description

This utility function gives a summary table of a belief function with its mass function and the associated measures of belief and plausibility.

Usage

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

Arguments

`x`	A belief function in its bca form, generally the normalized result of the combination of two or more belief functions (see `nzdsr`.
`singletonsOnly`	= TRUE reduces the table of results to elementary events (singletons).
`removeZeroes`	= TRUE removes focal elements with 0 mass.

Value

A list of three elements:

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

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, infovarnames = "xy")
xyNorm <- nzdsr(xy, infovarnames = "xyNorm")
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

addTobca, 2
aplDecode *(decode)*, 7
aplEncode *(encode)*, 12
aplRDV *(reduction)*, 19
aplShape *(shape)*, 20

bca, *2*, 3, *6*, *10*, *16*
bcaRel, *4*, 4
belplau, 6
bpa *(bca)*, 3

decode, 7
dotprod, 8
doubles, 9
dsrwon, 10, *16*

elim, 11
encode, 12
extmin, 13

inters, 14

nameRows, *4*, 15
nzdsr, *10*, 16, *17*, *21*

plautrans, 17
productSpace, *3*, 18

reduction, 19

shape, 20

tabresul, 21