

Package: dst (via r-universe)

October 12, 2026

Type Package

Title Using the Theory of Belief Functions

Encoding UTF-8

Version 1.9.0

Date 2024-08-24

Description Using the Theory of Belief Functions for evidence calculus. Basic probability assignments, or mass functions, can be defined on the subsets of a set of possible values and combined. 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)

LazyData true

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

RoxygenNote 7.3.3

Depends R ($\geq 3.5.0$)

Imports dplyr, ggplot2, tidyr, Matrix, methods, parallel, rlang, utils, bit, progress

Suggests igraph, knitr, rmarkdown, tidyverse, testthat

VignetteBuilder knitr

Config/pak/sysreqs libicu-dev

Repository <https://rapler.r-universe.dev>

Date/Publication 2026-10-12 19:18:18 UTC

RemoteUrl <https://github.com/rapler/dst-1>

RemoteRef HEAD

RemoteSha 21d3b621e0b53e75d492a94f31996108677b6352

Contents

addToDSM	3
ads	4
captain_result	5
commonality	5
decode	6
dlfm	7
DoSSnames	8
dotprod	9
dproj	10
DSA	11
DSAh	12
DSC	13
DSM	15
encode	18
extSSR	19
fw	20
inters	21
intersBySSName	22
jointDSM	23
marrayToMatrix	25
matrixToMarray	26
mrf	27
mrt	28
nameCols	29
nameCols_prod	29
nameRows	30
normalize	31
peeling	32
printDSM	34
productSpace	35
PT	36
PTsimDSM	37
shape	38
SSR	39
swr	40
tabresul	41
truncateDSM	42
ttmatrix	43
uproj	44
Index	46

addToDSM	<i>Add some elements of 0 mass to an existing DSM.</i>
----------	--

Description

You may want to add to a previously defined DSM some elements or some subsets of the state space (the set of possible values). The mass of these additional elements or subsets may be set to zero. This feature is useful when one wants to examine the plausibility measure of elements or subsets of zero mass.

Usage

```
addToDSM(x, tt)
```

```
addTobca(x, tt)
```

Arguments

x	A DSM (see DSM).
tt	A matrix constructed in a boolean style (0,1) or a boolean matrix. The number of columns of the matrix tt must match the number of columns of the tt matrix of x (see DSM). Each row of the matrix identify a subset of the state space.

Value

x The original DSM **x** augmented with the added subsets defined by **tt**.

Author(s)

Claude Boivin

Examples

```
y <- DSM(tt = matrix(c(1,0,0,1,1,1),nrow=2, byrow = TRUE),
m = c(0.6, 0.4), cnames = c("a", "b", "c"), idvar = 1)
addToDSM(y, matrix(c(0,1,0,0,0,1, 0,1,1), nrow = 3, byrow = TRUE))
x <- DSM(tt = 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"), idvar = 1)
xy <- dsrwon(x,y)
xy1 <- addToDSM(nzdsr(xy), matrix(c(0,1,0,0,0,1), nrow = 2, byrow = TRUE))
xy1
# add all singletons to a DSM
addToDSM(x, tt = diag(rep(1, ncol(x$tt) ) ) )
```

ads	<i>The Captain's Problem. ads: Relation between variables Arrival (A), Departure delay (D) and Sailing delay (S)</i>
-----	--

Description

This dataset is the **tt** matrix establishing the relation $A = D + S$, where $A = 0:6$, $D = 0:3$ and $S = 0:3$. The subset made of all the triplets (a,d,s) of ($A \times D \times S$) where $a = d + s$ is true has a mass value of 1. To construct the **tt** matrix, we put the variables A, D, S side by side, as in a truth table representation. Each triplet of the subset is described by a row of the matrix as a vector of zeros and ones.

Usage

ads

Format

An integer matrix with 18 rows and 17 columns

[1, c(1,2)] value = 0, not used

[1, 3:17] Identification numbers of the three variables. Column 3 to 9: variable 1; column 10 to 13: variable 2; column 14 to 17: variable 3.

nospec identification number of the specification

m the value of the specification, a number between 0 and 1

6 1 if 6 is part of the specification, 0 otherwise

5 1 if 5 is part of the specification, 0 otherwise

4 1 if 4 is part of the specification, 0 otherwise

3 1 if 3 is part of the specification, 0 otherwise

2 1 if 2 is part of the specification, 0 otherwise

1 1 if 1 is part of the specification, 0 otherwise

0 1 if 0 is part of the specification, 0 otherwise

Author(s)

Claude Boivin, Stat.ASSQ

Source

Almond, R.G. [1988] Fusion and Propagation in Graphical Belief Models. Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface. Wegman, Edward J., Gantz, Donald T. and Miller, John J. (ed.). American Statistical Association, Alexandria, Virginia. pp 365-370.

captain_result	<i>The Captain's Problem.</i> swr: <i>Result of the evaluation of the Hypergraph at node Arrival (A)</i>
----------------	---

Description

This dataset is the `tt` bca resulting from the combination of the relations of the hypergraph and marginalization at node Arrival (A).

Usage

```
captain_result
```

Format

A list of 8 elements, of class `bcaspec`.

Author(s)

Claude Boivin, Stat.ASSQ

Source

Almond, R.G. [1988] Fusion and Propagation in Graphical Belief Models. Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface. Wegman, Edward J., Gantz, Donald T. and Miller, John J. (ed.). American Statistical Association, Alexandria, Virginia. pp 365–370.

commonality	<i>Compute qq from tt</i>
-------------	---------------------------

Description

`qq` is the commonality function as a set function from the subsets of the frame to $[0, 1]$. To evaluate it, input a set encoded in binary vector, so the commonality number at that set can be returned.

Usage

```
commonality(tt, m, method = NULL)
```

Arguments

<code>tt</code>	Boolean description matrix
<code>m</code>	Mass assignment vector of probabilities
<code>method</code>	= <code>NULL</code> : Use Fast Zeta Transform ("fzt") or Efficient Zeta Transform ("ezt") or Efficient Zeta Transform on a meet-closed subset ("ezt-m")

Value

f Commonality function

Author(s)

Peiyuan Zhu

Examples

```
x <- bca(tt = 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"), varnames = "x", idvar = 1)
qq <- commonality(x$tt,x$spec[,2], method = "ezt")
qq
qq1 <- commonality(x$tt,x$spec[,2])
qq1
```

decode

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

Description

The `ap1Decode` 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, 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

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
```

dlfm

The Captain's Problem. dlfm: Relation between variables Departure delay (D), Loading delay (L), Forecast of the weather (F), Maintenance delay (M)

Description

This dataset is the **tt** matrix establishing the relation between the four variables. Each event (loading = true, forecast = foul, Maintenance = true) adds one day of Departure Delay. The elements (d,l, f, m) of (D x L x F x M) satisfying the relation form a subset with a mass value of 1. To construct the **tt** matrix, we put the variables D,L,F,M side by side, as in a truth table representation. Each 4-tuple of the subset is described by a row of the matrix as a vector of zeros and ones.

Usage

dlfm

Format

An integer matrix with 10 rows and 12 columns.

[1,c(1,2)] value = 0, not used

[1,3:12] Identification numbers of the four variables. Column 3 to 6: variable 2; columns 7,8: variable 4; columns 9, 10: variable 5; columns 11,12: variable 6.

nospec identification number of the specification

m the value of the specification, a number between 0 and 1

3 1 if d3 is part of the specification, 0 otherwise

2 1 if d2 is part of the specification, 0 otherwise

1 1 if d1 is part of the specification, 0 otherwise
0 1 if d0 is part of the specification, 0 otherwise
true 1 if true is part of the specification, 0 otherwise
false 1 if false is part of the specification, 0 otherwise
foul 1 if foul is part of the specification, 0 otherwise
fair 1 if fair is part of the specification, 0 otherwise

Author(s)

Claude Boivin, Stat.ASSQ

Source

Almond, R.G. [1988] Fusion and Propagation in Graphical Belief Models. Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface. Wegman, Edward J., Gantz, Donald T. and Miller, John J. (ed.). American Statistical Association, Alexandria, Virginia. pp 365–370.

DoSSnames

Construct subsets names from column names of a tt matrix

Description

Construct subsets names from column names of a tt matrix

Usage

DoSSnames(tt)

Arguments

tt A description matrix with column names

Value

subsets_names A list of names.

Author(s)

Claude Boivin

Examples

```
y1 <- bca(tt = 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"),
varnames = "x", idvar = 1)
DoSSnames(y1$tt)
```

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

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))
```

dproj	<i>Reduction of a jointDSM</i>
-------	--------------------------------

Description

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

Usage

```
dproj(rel, xnb)
```

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

Arguments

rel	The jointDSM to reduce, an object of class DSMspec.
xnb	Identification number of the variable to eliminate.

Value

r The reduced jointDSM

Author(s)

Claude Boivin

Examples

```
# We construct a jointDSM between two variables to show marginalization.
wr_tt <- matrix(c(1,rep(0,3),rep(c(1,0),3),0,1,1,1,0,0,
1,0,rep(1,5),0,1,1,0,rep(1,5))), ncol = 4, byrow = TRUE)
colnames(wr_tt) <- c("Wy Ry", "Wy Rn", "Wn Ry", "Wn Rn")
rownames(wr_tt) <- nameRows(wr_tt)
wr_spec = matrix(c(1:8, 0.017344, 0.046656,
0.004336, 0.199456,0.011664,0.536544,0.049864, 0.134136),
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, varnames = c("Roadworks","Rain"),
valuenames = list( RdWorks = c("Wy", "Wn"), Rain=c("Ry", "Rn") ))
class(wr_rel) <- "DSMspec"
DSMprint(dproj(wr_rel, xnb = 5))
DSMprint(dproj(wr_rel, xnb = 4))
```

DSA	<i>Calculate the degrees of unambiguous support (p), unambiguous contradiction (q), and ambiguity (r) of a list of hypothesis based on a Dempster-Shafer Model (DSM).</i>
-----	--

Description

Degrees of unambiguous support p and unambiguous contradiction q of the focal sets of a DSM are computed. The ratio of the plausibility of a focal set against the plausibility of its contrary is also computed. Subsets with zero mass can be excluded from the calculations.

Usage

```
DSA(x, remove = FALSE, h = NULL)
```

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

Arguments

<code>x</code>	A DSM (see DSM).
<code>remove</code>	= TRUE: Exclude subsets with zero mass.
<code>h</code>	= NULL: Hypothesis to be tested. Description matrix in the same format than <code>x</code>

Details

The degree of unambiguous support (belief) p is defined by:

$$p(A) = \text{Sum}((m(B); B \subseteq A))$$

for every subset B of A .

The degree of non-contradiction (plausibility) $plau$ is defined by:

$$plau(A) = \text{Sum}[(m(B); B \cap A \neq \emptyset)]$$

for every subset B of the state-space model.

The plausibility ratio of a focal set A versus its contrary $\text{not } A$ is defined by: $(1 - q(A)) / (1 - p(A))$.

Value

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

- Column 1: the degree of unambiguous support (Belief) p ;
- Column 2: the degree of unambiguous contradiction (Disbelief) q ;
- Column 3: the degree of ambiguity (Epistemic uncertainty) r ;
- Column 4: the degree of non-contradiction (Plausibility) $plau$;
- Column 5: the Plausibility ratio $rplau$.

Author(s)

Claude Boivin, Peiyuan Zhu

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.
- Dempster, A., (2008). The Dempster-Shafer calculus for statisticians. International Journal of approximate reasoning, 48(2), 365-377.

Examples

```
x <- DSM(tt = 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"), varnames = "x", idvar = 1)
DSA(x)
y <- DSM(tt = matrix(c(1,0,0,1,1,1),nrow = 2,
byrow = TRUE), m = c(0.6, 0.4),
cnames = c("a", "b", "c"), varnames = "y", idvar = 1)
xy <- normalize(DSC(x,y))
DSA(xy)
print("compare all elementary events")
xy1 <- addToDSM(x = xy, tt = matrix(c(0,1,0,0,0,1), nrow = 2, byrow = TRUE))
DSA(xy1)
DSA(xy1, remove = TRUE)
DSA(xy1, h = matrix(c(1,0,0,0,1,1), nrow = 2, byrow = TRUE))
```

DSAh

Calculate degree of unambiguous support (p), unambiguous contradiction (q), ambiguity (r), plausibility, and plausibility ratio

Description

Calculate degree of unambiguous support (p), unambiguous contradiction (q), ambiguity (r), plausibility, and plausibility ratio

Usage

DSAh(MACC, W2, h)

belplauH(MACC, W2, h)

Arguments

MACC	Vector of masses e.g. <code>x\$spec[,2]</code>
W2	Description matrix e.g. <code>x\$tt</code>
h	Hypotheses to be tested, same format as <code>x\$tt</code>

Value

A matrix of `M` rows by 5 columns is returned, where `M` is the number of hypothesis tested:

- Column 1: the degree of unambiguous support `p`;
- Column 2: the degree of unambiguous contradiction `q`;
- Column 3: the degree of ambiguity `r`;
- Column 4: the degree of Plausibility `plau`;
- Column 5: the Plausibility ratio `rplau`.

Author(s)

Peiyuan Zhu

Examples

```
x <- DSM(tt = matrix(c(1,1,0,1,1,1), nrow = 2, byrow = TRUE), m = c(0.8, 0.2), cnames = c(1,2,3))
DSAh(MACC = x$spec[,2], W2 = x$tt, h = x$tt)
hyp <- matrix(c(0,1,0,0,1,1), nrow = 2, byrow = TRUE)
rownames(hyp) <- nameRows(hyp)
DSAh(MACC = x$spec[,2], W2 = x$tt, h = hyp)
```

DSC

Combination of two Dempster-Shafer models

Description

The unnormalized Dempster's rule is used to combine two mass functions `mx` and `my` defined on the same state-space representation (SSR) and described by their respective basic mass 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 function [normalize](#)).

Usage

```
DSC(
  x,
  y,
  mcores = "no",
  use_ssnames = FALSE,
  use_sparse = "no",
```

```

    varnames = NULL,
    skpt_tt = FALSE,
    relnb = NULL,
    infovarnames
)

dsrwon(
  x,
  y,
  mcores = "no",
  use_ssnames = FALSE,
  use_sparse = "no",
  varnames = NULL,
  skpt_tt = FALSE,
  relnb = NULL,
  infovarnames
)

```

Arguments

x	A Dempster-Shafer model (see DSM).
y	A Dempster-Shafer model (see DSM).
mcores	Make use of multiple cores ("yes") or not ("no"). Default = "no".
use_ssnames	= TRUE to use ssnames instead of tt matrix to do the intersections. Default = FALSE
use_sparse	Make use of sparse matrices ("yes") or not ("no"). Default = "no".
varnames	A character string to name the resulting variable. named "z" if omitted.
skpt_tt	Skip reconstruction of tt matrix. Default = FALSE.
relnb	Identification number of the jointDSM. Can be omitted.
infovarnames	Deprecated. Old name for varnames .

Details

The two DSM's **x** and **y** must be defined on the same SSR for the combination to take place. The jointDSM number of the x input is given to the output result.

Value

A Dempster-Shafer model with these two components added:

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

Author(s)

Claude Boivin, Peiyuan Zhu

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
- Dempster, A., (2008). The Dempster–Shafer calculus for statisticians. International Journal of approximate reasoning, 48(2), 365-377.

Examples

```

y1 <- DSM(tt = 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"),
varnames = "x", idvar = 1)
y2 <- DSM(tt = matrix(c(1,0,0,1,1,1),nrow = 2,
byrow = TRUE), m = c(0.6, 0.4),
cnames = c("a", "b", "c"),
varnames = "x", idvar = 1)
DSC(y1,y2)
# Sparse matrices
y1s <- y1
y2s <- y2
y1s$tt <- methods::as(y1$tt, "RsparseMatrix")
y2s$tt <- methods::as(y2$tt, "RsparseMatrix")
y1y2s <- DSC(y1s, y2s, use_ssnames = TRUE)
vacuous <- DSM(matrix(c(1,1,1), nrow = 1), m = 1, cnames = c("a","b","c"))
DSC(vacuous, vacuous)

```

DSM

Unidimensional Dempster-Shafer model

Description

Function DSM 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 state-space representation (SSR) or "frame of discernment". Each subset A of Θ with a positive mass value is called a focal set. The associated mass value is a number of the $(0,1]$ interval, called "basic mass 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 value of zero.

Usage

```

DSM(
  tt = NULL,
  m,
  include_all = FALSE,

```

```

    ssr = NULL,
    cnames = NULL,
    con = NULL,
    ssnames = NULL,
    idvar = NULL,
    infovar = NULL,
    varnames = NULL,
    valuenames = NULL,
    inforel = NULL
)

bca(
  tt = NULL,
  m,
  include_all = FALSE,
  ssr = NULL,
  cnames = NULL,
  con = NULL,
  ssnames = NULL,
  idvar = NULL,
  infovar = NULL,
  varnames = NULL,
  valuenames = NULL,
  inforel = NULL
)

```

Arguments

tt	Mandatory. A (0,1)-matrix or a boolean matrix. The number of columns must match the number of elements (values) of the SSR Θ . Each row is a subset of Θ . The last row is the frame Θ , represented by a vector of 1's.
m	A numeric vector of length equal to the number of rows of the matrix tt . Values of m must lie in the interval $(0,1]$ and must add to one. The mass $m(k)$ represents the mass value allotted to the focal set represented by the row k of the matrix tt .
include_all	Default = FALSE. Put TRUE to include all elements with 0 mass in the DSM.
ssr	State space representation. Contains all the information pertaining to the underlying state space. See SSR .
cnames	A character vector containing the names of the elements of the SSR Θ . The length must be equal to the number of elements of Θ . The names are first searched in the valuenames parameter. If NULL, column names of the matrix tt are taken if present. Otherwise, names are generated.
con	The measure of conflict can be provided. 0 by default.
ssnames	A list of subsets names which will be obtained from the column names of the tt matrix.

<code>idvar</code>	The number given to the variable. A number is necessary to manage relations between variables of multidimensional state spaces and enable computations on a graph. 0 if omitted.
<code>infovar</code>	A two-column matrix containing variable identification number and the number of elements of the variable. Generated if omitted.
<code>varnames</code>	The name of the variable. Generated if omitted.
<code>valuenames</code>	A list of the names of the variables with the name of the elements of their SSR.
<code>inforel</code>	Not used here. Defined within function jointDSM .

Details

There is two ways of defining the DSM: a (0,1) matrix or a list of subsets labels.

Value

y An object of class `DSMspec` called a DSM for "Dempster-Shafer model":

- `tt` The table of focal sets. Rownames of the matrix of focal sets are generated from the column names of the elements of the frame. See [nameRows](#) for details.
- `spec` A two column matrix. First column contains numbers given to the subsets, 1 to `nrow(tt)`. Second column contains the mass values of the subsets.
- `con` The measure of conflict.
- `infovar` The number of the variable and the size of the SSM.
- `varnames` The name of the variable.
- `valuenames` A list of length 1 consisting of the name of the variable with the names of the elements of the SSM (the column names of the `tt` matrix).
- `ssnames` A list of subsets names done from the column names of the `tt` matrix.
- `inforel` Set at 0. used in function [jointDSM](#).

Author(s)

Claude Boivin

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
- Dempster, A., (2008). The Dempster–Shafer calculus for statisticians. International Journal of approximate reasoning, 48(2), 365-377.

Examples

```
tt<- t(matrix(c(1,0,1,1),ncol = 2))
m<- c(.9,.1)
cnames <- c("yes","no")
DSM(tt, m)
DSM(tt, m, cnames=cnames)
tt1<- t(matrix(c(1,0,1,1),ncol = 2))
colnames(tt1) <- c("yes", "no")
m <- c(.9, .1)
DSM(tt=tt1, m, idvar = 1)
y <- DSM(tt=matrix(c(1,0,0,1,1,1),nrow = 2,
byrow = TRUE), m = c(0.6,0.4), include_all = TRUE,
cnames = c("a", "b", "c"),varnames = "y", idvar = 1)
printDSM(y)
vacuous <- DSM(matrix(c(1,1,1), nrow = 1), m = 1, cnames = c("a","b","c"), ssnames = c("a","b","c"))
printDSM(vacuous)
ss_Theta <- SSR(varnames = "x", idvar = 1, size = 3, cnames = c("a", "b", "c"))
X <- DSM(tt=matrix(c(1,0,0,0,1,0,0,0,1), nrow = 3, byrow = TRUE), m = c(0.2, 0.3, 0.5), ssr = ss_Theta)
printDSM(X)
```

encode

Convert a value to its representation in another chosen base

Description

The `ap1Encode` 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 recode 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

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.
```

extSSR	<i>Extension of a State Space Representation (SSR)</i>
--------	--

Description

This function works on a marginal DSM x defined on a single variable. It allows the addition of new values to the SSR.

Usage

```
extSSR(x, use_ssnames = FALSE, lab = NULL)

extFrame(x, use_ssnames = FALSE, lab = NULL)
```

Arguments

x	An object of DSM class, i.e. a basic chance assignment defined on one SSR.
use_ssnames	Default= FALSE. Put TRUE if use of subset names is wanted.
lab	A character vector containing the names of the elements to add to the SSR.

Value

zxtn The DSM within its SSR extended.

Author(s)

Claude Boivin

Examples

```
s1_e1 <- DSM(tt = matrix(c(1,0,1,1),nrow = 2, byrow = TRUE),
m = c(0.6,0.4), cnames = c("S1","S2"), varnames = "v1", idvar = 1)
s13_names <- extSSR(s1_e1, lab = "S3", use_ssnames =TRUE)
s13 <- extSSR(s1_e1, lab = "S3")
s13$valuenames
```

fw	<i>The Captain's Problem. fw: Relation between variables Forecast of the weather (F) and Weather at sea (W)</i>
-----------	---

Description

This dataset is the **tt** matrix establishing the relation between the two variables. An accurate forecast is described by this subset of two event: (Forecast = foul, Weather = foul) and (Forecast = fair, Weather = fair). We assign a mass value of 0.8 to this subset. The remaining mass of 0.2 is allotted to the frame. To construct the **tt** matrix, we put the variables F and W side by side, as in a truth table representation. Each pair of the subset is described by a row of the matrix as a vector of zeros and ones.

Usage

fw

Format

An integer matrix with 4 rows and 6 columns.

[1,c(1,2)] value = 0, not used

[1,3:6] Identification numbers of the two variables. Column 3,6: variable 5; columns 5,6: variable 7.

nospec identification number of the specification

m the value of the specification, a number between 0 and 1

foul 1 if foul is part of the specification, 0 otherwise

fair 1 if fair is part of the specification, 0 otherwise

Author(s)

Claude Boivin

Source

Almond, R.G. [1988] Fusion and Propagation in Graphical Belief Models. Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface. Wegman, Edward J., Gantz, Donald T. and Miller, John J. (ed.). American Statistical Association, Alexandria, Virginia. pp 365–370.

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

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 <- matrix(c(FALSE, TRUE, TRUE), nrow=1)
b2 <- matrix(c(TRUE, TRUE, FALSE), nrow=1)
colnames(b1) <- colnames(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)
# Sparse matrices
stt1 <- Matrix::sparseMatrix(i= c(1,1,2,2,3,3,3), j= c(2,3,1,2,1,2,3), x = 1, dims = c(3,3))
```

```

y1 <- bca(tt = stt1, m = c(0.2,0.5, 0.3),
          cnames = c("a", "b", "c"),
          varnames = "x", idvar = 1)
stt2 <- Matrix::sparseMatrix(i= c(1,2,2,2), j= c(1,1,2,3), x = 1, dims = c(2,3))
y2 <- bca(tt = stt2, m = c(0.6, 0.4),
          cnames = c("a", "b", "c"),
          varnames = "x", idvar = 1)
sr <- inters(y1$tt, y2$tt)
sr
class(sr)

```

intersBySSName	<i>Intersect two vectors of ssnames</i>
----------------	---

Description

Intersect two vectors of ssnames

Usage

```
intersBySSName(zx, yz)
```

Arguments

zx	a vector of ssnames from one bca
yz	a vector of ssnames from another bca

Value

ssnames in the intersection of the two bcas

Author(s)

Peiyuan Zhu

Examples

```

y1 <- bca(tt = 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"),
varnames = "x", idvar = 1)
y2 <- bca(tt = matrix(c(1,0,0,1,1,1),nrow = 2,
byrow = TRUE), m = c(0.6, 0.4),
cnames = c("a", "b", "c"),
varnames = "x", idvar = 1)
intersBySSName(y1$ssnames[[1]], y2$ssnames[[2]])

```

jointDSM

*Multidimensional DS Model***Description**

Function `jointDSM` is used to assign mass values to subsets defined in a finite multidimensional state space P , which is the product space of unidimensional spaces $\Theta_1, \Theta_2, \dots, \Theta_n$. To achieve the definition of the joint DSM of a focal element, a relation between two or more unidimensional SSR is described by one or more subsets of $\Theta_1, \Theta_2, \dots, \Theta_n$. Hence each subset can possibly include more than one element. Complete disjunctive coding is used to represent a subset in the input matrix of the function.

Usage

```
jointDSM(
  tt,
  spec,
  infovar = NULL,
  ssrlist = NULL,
  varnames,
  valuenames,
  relnb = NULL
)

bcaRel(
  tt,
  spec,
  infovar = NULL,
  ssrlist = NULL,
  varnames,
  valuenames,
  relnb = NULL
)
```

Arguments

tt The description matrix of the subsets establishing the relation between $\Theta_1, \Theta_2, \dots, \Theta_n$. This matrix is made of one or more subsets and the whole SSR in the last row. The columns of the matrix are defined by putting the SSRs side by side, ordered by their `*idvar*` from left to right. The first columns are the labels of the first SSR, followed by the labels of the second SSR till the end. For each SSR, there are as many columns as there is possible values. A subset is then defined in the way of a truth table representation, where we keep the "TRUE" part of the table. Hence each row of the table is an element of a subset, described by a sequence of 0 (absence of value) or 1 (presence of value). This is called a complete disjunctive coding. CAUTION: SSRs put side by side must be ordered by their `*idvar*` from left to right.

spec	A two column matrix. First column: numbers assigned to the subsets. Second column: the mass values of the subsets. If a subset has more than one element (row), the number of the subset and its associated mass value are repeated to match the number of elements (rows) of the subset.
infovar	A two column matrix containing SSR identification numbers and the number of elements of each SSR. The identification numbers must be ordered in increasing number. May be omitted if the state space list is given (parameter <i>*ssrlist*</i>).
ssrlist	The list of 1D state spaces to use.
varnames	The names of the SSR. May be omitted if the state space list is given (parameter <i>*ssrlist*</i>).
valuenames	A list of the names of the SSR with the name of the elements of their SSR. May be omitted if the state space list is given (parameter <i>*ssrlist*</i>).
relnb	A number given to the jointDSM. Set at 0 if omitted.

Value

An object of class `DSMspec` called a DSM or "basic mass assignment". This is a list containing the following components:

- `con` The measure of conflict.
- `tt` The resulting table of subsets in its joint state Space representation. Rownames of the matrix of subsets are generated from the column names of the elements of the product space. See [nameRows](#) for details.
- `spec` The resulting two-column matrix of specification numbers with associated mass values.
- `infovar` The two-column matrix of SSR number and size.
- `valuenames` A list of the names of the SSR with the name of their elements.
- `inforel` A two-column matrix containing the relation number and the number of SSRs of the relation.

Author(s)

Claude Boivin

Examples

```
# A logical implication rule
# A typical relation between two variables is the
# logical implication a -> b. Let us suppose
# that a stands for Rain: {yes, no} and b stands for
# Roadworks: {yes, no}. From experience,
# I am 75 % sure that there will be RoadWorks if there is no rain.

# 0. Define State space representations
ssWorks <- SSR(varnames = "W", idvar = 4, size = 2, cnames = c("Wy", "Wn") )
ssRain <- SSR(varnames = "R", idvar = 5, size = 2, cnames = c("Ry", "Rn") )
```

```

ssrWR <- list(ssWorks, ssRain)

# 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, unlist(sapply(ssrWR, `[`,4))) )

# 2. 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")))

rel <- jointDSM(tt = ttrwf, spec = specrw, ssrlist = ssrWR, relnb = 6)

```

marrayToMatrix

Transformation of an array data to its matrix representation

Description

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

Usage

```
marrayToMatrix(mtt)
```

Arguments

mtt The matrix **tt** of the relation in array format

Value

The matrix representation of the data.

Author(s)

Claude Boivin

Examples

```

mtt <- array(c(1,0,0,0,1,1,0,0,1,0,0,1,1,0,1,0,1,1,0,1,1,1,0,1,0,1,1,1,1,1), c(2,2,8),
dimnames = list( RdWorks=c("Wy", "Wn") , Rain = c("Ry", "Rn"), ev=1:8) )
print(z <- marrayToMatrix(mtt))

```

matrixToMarray	<i>Transformation of the tt matrix of a relation</i>
----------------	--

Description

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

Usage

```
matrixToMarray(tt, valuenames)
```

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.
valuenames	A list of the names of the variables with the name of each value of their frame of discernment.

Value

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

Author(s)

Claude Boivin

Examples

```
# Define wr_tt, a matrix describing the relation between two variables
wr_tt <- matrix(c(1,rep(0,3),rep(c(1,0),3),0,1,1,1,0,0,
1,0,rep(1,5),0,1,1,0,rep(1,5))), ncol=4, byrow = TRUE)
colnames(wr_tt) <- c("Wy Ry", "Wy Rn", "Wn Ry", "Wn Rn")
rownames(wr_tt) <- nameRows(wr_tt)
vars = list( RdWorks = c("Wy", "Wn") , Rain = c("Ry", "Rn"))
print(zmToa <- matrixToMarray(tt = wr_tt, valuenames = vars ) )
```

mrf	<i>The Captain's Problem. mrf: Relation between variables No Maintenance ($M = \text{false}$) and Repairs at sea (R)</i>
------------	--

Description

This dataset is the **tt** matrix establishing a set of two relations between the two variables. First, Repairs = true if Maintenance = false in ($M \times R$). We are 20% sure that there will be Repairs if no maintenance. Second, Repairs = false if Maintenance = false in ($M \times R$). We are 20% sure that there will be no repairs if no maintenance.

Usage

mrf

Format

A (0,1) matrix with 4 rows and 6 columns.

[1,c(1,2)] value = 0, not used

[1,3:6] Identification numbers of the two variables. Column 3,4: variable 6; columns 5,6: variable 8

nospec identification number of the specification

m the value of the specification, a number between 0 and 1

true 1 if true is part of the specification, 0 otherwise

false 1 if false is part of the specification, 0 otherwise

Details

These two relations are implication rules. The remaining mass of 0.6 is allotted to the frame. To construct the **tt** matrix, we put the variables M and R side by side, as in a truth table representation. Each pair of the subset is described by a row of the matrix as a vector of zeros and ones.

Author(s)

Claude Boivin, Stat.ASSQ

Source

Almond, R.G. [1988] Fusion and Propagation in Graphical Belief Models. Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface. Wegman, Edward J., Gantz, Donald T. and Miller, John J. (ed.). American Statistical Association, Alexandria, Virginia. pp 365–370.

mrt

The Captain's Problem. mrt: Relation between variables Maintenance done ($M = \text{true}$) and Repairs at sea (R)

Description

This dataset is the **tt** matrix establishing a set of two relations between the two variables. First, Repairs = true if Maintenance = true in ($M \times R$). We are 10% sure that there will be Repairs if maintenance is done. Second, Repairs = false if Maintenance = true in ($M \times R$). We are 70% sure that there will be no repairs if maintenance is done.

Usage

```
mrt
```

Format

A (0,1) matrix with 4 rows and 6 columns.

[1,c(1,2)] value = 0, not used

[1,3:6] Identification numbers of the two variables. Column 3,4: variable 6; columns 5,6: variable 8

nospec identification number of the specification

m the value of the specification, a number between 0 and 1

true 1 if true is part of the specification, 0 otherwise

false 1 if false is part of the specification, 0 otherwise

Details

These two relations are implication rules. The remaining mass of 0.2 is allotted to the frame. To construct the **tt** matrix, we put the variables M and R side by side, as in a truth table representation. Each pair of the subset is described by a row of the matrix as a vector of zeros and ones.

Author(s)

Claude Boivin, Stat.ASSQ

Source

Almond, R.G. [1988] Fusion and Propagation in Graphical Belief Models. Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface. Wegman, Edward J., Gantz, Donald T. and Miller, John J. (ed.). American Statistical Association, Alexandria, Virginia. pp 365–370.

nameCols	<i>Naming the columns of the <code>tt</code> matrix</i>
----------	---

Description

This utility function makes use of the `valuenames` and `size` parameters of a set of variables to assign values names to the columns of a `tt` matrix.

Usage

```
nameCols(valuenames, size)
```

Arguments

valuenames	A list of the names of the variables with the name of the elements of their frame of discernment.
size	A vector of the size of the variables.

Value

A character vector of length equal to the sum of the sizes of the variables.

Author(s)

Claude Boivin

Examples

```
infoval <- list(A = c("a1", "a2"), B = c("b1", "b2", "b3"))
sizes <- c(2,3)
print(nameCols(valuenames = infoval, size = sizes) )
```

nameCols_prod	<i>Naming the columns of the <code>tt</code> matrix of a product space</i>
---------------	--

Description

This utility function makes use of the `valuenames` and `size` parameters of a set of variables to assign values names to the columns of the `tt` matrix of their product space.

Usage

```
nameCols_prod(valuenames, size)
```

Arguments

- valuenames** A list of the names of the variables with the name of the elements of their frame of discernment.
- size** A vector of the size of the variables.

Value

A character vector of length equal to the product of the sizes of the variables.

Author(s)

Claude Boivin

Examples

```
infoval <- list(A = c("a1", "a2"), B = c("b1", "b2", "b3"))
sizes <- c(2,3)
print(nameCols_prod(valuenames = infoval, size = sizes) )
```

nameRows	<i>Combining the column names of a matrix to construct names for the rows</i>
-----------------	---

Description

This function determines the name of a row from all the columns of the **tt** that show 1 for that row.

Usage

```
nameRows(tt, f)
```

Arguments

- tt** A (0,1)-matrix or a boolean matrix.
- f** Deprecated. Old name for **tt** matrix.

Details

The row containing only 1s is called "frame", to avoid too long a label. The empty set is identified 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 can become very long.

Value

A character vector of labels obtained for the rows of the **tt** matrix. The length of the result is **nrow(tt)**.

Author(s)

Claude Boivin

Examples

```
tt <- matrix(c(0,0,0,1,0,0,0,0,1,1,0,1,1,1,1),ncol = 3, byrow = TRUE)
colnames(tt) <- c("A","B","C")
rownames(tt) <- nameRows(tt)
tt
```

normalize

Normalization of a basic chance assignment

Description

It may occur that the result of the combination of two basic chance assignments with Dempster's Rule of combination contains a non-zero mass allocated to the empty set. The function **nzdsrc** 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

```
normalize(x, sparse = "no", comm = "no")

nzdsrc(x, sparse = "no", comm = "no")
```

Arguments

x	A basic chance assignment, i.e. a object of class <code>bcaspec</code> .
sparse	Put "yes" to use sparse matrix. Default = "no".
comm	Put "yes" to use commonality function. Default = "no".

Value

z The normalized basic chance assignment.

Author(s)

Claude Boivin, Peiyuan Zhu

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(tt= matrix(c(1,0,1,1),nrow = 2, byrow = TRUE),
m = c(0.9,0.1), cnames = c("yes", "no"),
varnames = "x", idvar = 1)
x2 <- bca(tt = matrix(c(0,1,1,1),nrow = 2, byrow = TRUE),
m = c(0.5,0.5), cnames = c("yes", "no"),
varnames = "x", idvar = 1)
print("combination of x1 and x2")
x1x2 <- dsrwon(x1,x2, varname = "x")
nzdsr(x1x2)
# Test with sparse matrices
x1s=x1
x2s=x2
x1s$tt <- methods::as(x1$tt, "RsparseMatrix")
x2s$tt <- methods::as(x2$tt, "RsparseMatrix")
x1x2s <- dsrwon(x1s, x2s, use_ssnames = TRUE)
nzdsr(x1x2s)

print("normalization of a bca definition.")
y2 <- bca(tt = 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"), idvar = 1)
cat("y2")
cat("\  ")
y2
nzdsr(y2)
```

peeling

The peeling algorithm

Description

An implementation of the peeling algorithm based on its description in terms of hypergraphs by R. Almond [1989].

Usage

```
peeling(vars_def, hgm, hg_rel_names, elim_order, verbose = FALSE)
```

Arguments

vars_def	A list of variables and their possible values. Concatenate the valuenames parameter of all the variables of the hypergraph to obtain this list.
hgm	The incidence matrix of the hypergraph (bipartite graph), which is the description of the relations between the variables. The variables are the nodes of the hypergraph, and the relations are the edges. Each column describes a relation between the variables by a (0,1) vector. A "1" indicates that a variable belongs to the relation and a "0" not. This matrix

must have row and column names. These names are used to show the graph eventually. They need not be the same as variables and relations names of the set of bca's to be analyzed. Use short names to obtain a clear graph.

hg_rel_names The names of the relations, which are objects of class "bcaspec".

elim_order The order of elimination of the variables. A vector of length `nrow(hgm)`. Variables are identified by numbers. The first number gives the first variable to eliminate. The variable of interest comes last.

verbose = TRUE: print steps on the console. Default = FALSE.

Details

The peeling algorithm works on an undirected graph. Nodes of the graph (variables) are removed one by one until only the variable of interest remains. An order of elimination (peeling) of the variables must be chosen by the user. No algorithm is provided for that matter. At each step, a procedure of extension is applied to the bca's to merge, and marginalization is applied to eliminate a variable. The marginalization has the effect to integrate in the remaining nodes the information of the eliminated variable.

Value

A bca class object.

Author(s)

Claude Boivin

References

- Almond, R. G. (1989) Fusion and Propagation of Graphical Belief Models: An Implementation and an Example. Ph. D. Thesis, the Department of Statistics, Harvard University. 288 pages (for the description of the peeling algorithm, see pages 52-53).

Examples

```
# Zadeh's Example
# 1. Defining variables and relations
# (for details, see vignette: Zadeh_Example)
e1 <- bca(tt = matrix(c(1,0,0,1,1,1), ncol = 2, byrow = TRUE),
  m = c(0.99, 0.01, 0), cnames = c("M", "T"),
  varnames = "D1", idvar = 1)
e2 <- bca(tt = matrix(c(1,0,0,1,1,1), ncol = 2, byrow = TRUE),
  m = c(0.99, 0.01, 0), cnames = c("C", "T"),
  varnames = "D2", idvar = 2)
p_diag <- bca(tt = matrix(c(1,1,1), ncol = 3, byrow = TRUE),
  m = 1, cnames = c("M", "T", "C"),
  varnames = "D", idvar = 3)
# Defining the relation between the variables
# tt matrix
tt_r1 <- matrix(c(1,0,1,0,1,0,0,1,0,1,0,0,0,1,
```

```

1,0,0,1,1,0,0,1,0,0,1,0,1,0,0,1,1,0,0,1,0,
0,1,1,0,0,0,1,0,1,0,1,0,1,0,1,1,1,1,1,1,1),
ncol = 7,byrow = TRUE)
colnames(tt_r1) = c("M", "T", "C", "T", "M", "T", "C")
# The mass function
spec_r1 <- matrix(c(rep(1,7),2, rep(1,7), 0), ncol = 2, dimnames = list(NULL, c("specnb", "mass")))
# Variables numbers and dimension of their frame
info_r1 <- matrix(c(1:3, 2,2,3), ncol = 2, dimnames = list(NULL, c("varnb", "size"))) )
# The relation between e1, e2 and a patient p
r1 <- bcaRel(tt = tt_r1, spec = spec_r1, infovar = info_r1,
  varnames = c("D1", "D2", "D"), relnb = 1)

# 2. Setting the incidence matrix of the graph
rel1 <- 1*1:3 %in% r1$infovar[,1]
ev1 <- 1*1:3 %in% e1$infovar[,1]
ev2 <- 1*1:3 %in% e2$infovar[,1]
meddiag_hgm <- matrix(c(ev1,ev2, rel1), ncol = 3,
dimnames = list(c("D1", "D2", "D"), c("e1","e2", "r1")))

# 3. Setting the names of the variables and their frame of discernment
meddiag_vars1 <- c(e1$valuenames, e2$valuenames, p_diag$valuenames)

# 4. Names of bca specifications (evidence and relations)
meddiag_rel_names <- c("e1", "e2", "r1")

# 5. Order of elimination of variables
elim_order <- c(1,2,3)

tabresul(peeling(vars_def = meddiag_vars1, hgm = meddiag_hgm,
hg_rel_names = meddiag_rel_names, elim_order = c(1, 2, 3)) )

```

printDSM

Simple printing of the tt matrix and mass values of a Dempster-Shafer Model

Description

This utility function does a simple printing of a DSM (basic chance assignment).

Usage

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

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

Arguments

<code>x</code>	A list of class DSMspec.
<code>remove</code>	Default = FALSE. Put = TRUE to exclude subsets with zero mass.

Value

A table of subsets with their associated mass. Subsets are identified by row names.

Author(s)

Claude Boivin

Examples

```
z <- DSM(tt = 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"), idvar = 1)
printDSM(z)
y <- DSM(tt=matrix(c(1,0,0,1,1,1),nrow = 2,
byrow = TRUE), m = c(0.6,0.4), include_all = TRUE,
cnames = c("a", "b", "c"),varnames = "y", idvar = 1)
printDSM(y)
printDSM(y, remove = TRUE)
x <- DSM(tt=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"), idvar = 1, method = "ezt-m")
printDSM(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 variables are set side by side, as in a truth table. Each variable has a number of columns equal to the number of possible values.
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 by 1 or not. 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

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"))) )
rownames(ttfw) <- nameRows(ttfw)
ttfw
productSpace(tt = ttfw, specnb = specfw, infovar = infovarfw)
```

PT

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 this mass function.

Usage

PT(x)

plautrans(x)

Arguments

x A DSM.

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

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 <- DSM(tt = 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"),
varnames = "x", idvar = 1)
PT(x)
```

PTsimDSM	<i>Plausibility Transform for Combined Simple Dempster-Shafer Models (simDSM)</i>
----------	---

Description

Performs the plausibility transform on the combination of multiple simple Dempster-Shafer models (simDSMs) defined over a large state-space representation (SSR).

Usage

```
PTsimDSM(x, y, a, y0 = 0, flip = TRUE)
```

Arguments

x	A binary matrix where each row represents a focal element that is not the full SSR.
y	A logical column matrix indicating which rows of x may need inversion.
a	A numeric value ($0 \leq a \leq 1$) representing the mass assigned to each non-trivial focal element.
y0	A reference value (default = 0 or FALSE) used to determine which rows of x are inverted.
flip	Logical; whether to perform row inversion where applicable (default = TRUE).

Details

This function handles the case where several simple Dempster-Shafer models are combined on a very large SSR. Each row of the binary matrix **x** represents a focal element from one simDSM that is **not** the entire SSR. These focal elements can be specified directly or as complements, depending on which is more efficient to encode. A logical column matrix **y** identifies which rows of **x** need to be inverted (using the value **y0** as the check).

Value

A numeric plausibility vector `plau`.

Author(s)

Peiyuan Zhu

References

- Shafer, G., (1976). A Mathematical Theory of Evidence. Princeton University Press, Princeton, New Jersey, p. 38: Basic probability assignment.
- Barnett, J. A. (1991). Calculating dempster-shafer plausibility. IEEE transactions on pattern analysis and machine intelligence, 13(6), 599-602.
- 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

```
# Example usage:
x <- matrix(c(1, 0, 1, 1, 0, 0), nrow = 2, byrow = TRUE)
y <- matrix(c(TRUE, FALSE))
PTsimDSM(x, y, a = 0.3)
```

shape	<i>Obtain dimensions of an array or length of a vector with a single command</i>
--------------	--

Description

`shape` returns the dimension of given array or returns 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 dimension of the array **a** or the length of the vector **a**.

Author(s)

Claude Boivin

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"))
```

SSR

One dimensional State Space representation

Description

Function SSR is used to specify a set Θ of possible values.
The set Θ is called the state space.

Usage

```
SSR(varnames = NULL, idvar = NULL, size = NULL, cnames = NULL)
```

Arguments

varnames	The name of the variable. Generated if omitted.
idvar	The number given to the state space. A number is necessary to manage relations between variables of multidimensional state spaces and enable computations on a graph. 0 if omitted.
size	The number of elements of the state space representation Θ .
cnames	A character vector containing the names (labels) of the elements of the state space Θ . The length must be equal to the size parameter. If omitted, names are generated.

Value

y A list called a SSR for "State Space Representation":

- infovar The number of the variable and the size of the state space.
- varnames The name of the variable.
- valuenames A list of length 1 consisting of the name of the variable with the names (labels) of the elements of the state space.

Author(s)

Claude Boivin

References

- dDmpster (2007)...
- Gonng (2024)...

Examples

```
ss1 <- SSR("x1", 1,4,c("a","b","c","d"))
```

swr	<i>The Captain's Problem. swr: Relation between variables Sailing delay (S), Weather at sea (W), and Repairs at sea (R)</i>
-----	---

Description

This dataset is the **tt** matrix establishing a relation between S, W and R, where S = 0:3, W = (foul, fair) and R = (true, false). The goal of this relation is to account for other causes of sailing delay. All the elements (s,w,r) of (S x W x R) where W or R is true add one day of sailing delay. We put a mass value of 0.9 to this subset. The remaining mass of 0.1 is allotted to the frame.

Usage

```
swr
```

Format

An integer matrix with 6 rows and 10 columns.

[1,c(1,2)] value = 0, not used

[1,3:10] Identification numbers of the three variables. Column 3 to 6: variable 3; columns 7,8: variable 7, columns 9,10: variable 8

nospec identification number of the specification

m the value of the specification, a number between 0 and 1

3 1 if 3 is part of the specification, 0 otherwise

2 1 if 2 is part of the specification, 0 otherwise

1 1 if 1 is part of the specification, 0 otherwise

0 1 if 0 is part of the specification, 0 otherwise

foul 1 if foul is part of the specification, 0 otherwise

fair 1 if fair is part of the specification, 0 otherwise

true 1 if true is part of the specification, 0 otherwise

false 1 if false is part of the specification, 0 otherwise

Details

To construct the `tt` matrix, we put the variables S, W, R side by side, as in a truth table representation. Each triplet of the subset is described by a row of the matrix as a vector of zeros and ones.

Author(s)

Claude Boivin, Stat.ASSQ

Source

Almond, R.G. [1988] Fusion and Propagation in Graphical Belief Models. Computing Science and Statistics: Proceedings of the 20th Symposium on the Interface. Wegman, Edward J., Gantz, Donald T. and Miller, John J. (ed.). American Statistical Association, Alexandria, Virginia. pp 365–370.

<code>tabresul</code>	<i>Prepare a table of results</i>
-----------------------	-----------------------------------

Description

This utility function is a more detailed version of the `DSA` 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. Unlike function `DSA`, function `tabresul` does not reconstruct the row names from the column names. You can assign short rownames of your choice to the `tt` matrix of your resulting DSM before calling function `tabresul`.

Usage

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

Arguments

`x` A basic chance assignment (DSM)
`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

Examples

```

x <- DSM(tt = 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"),
varnames = "x", idvar = 1)
y <- DSM(tt = matrix(c(1,0,0,1,1,1),nrow = 2,
byrow = TRUE), m = c(0.6, 0.4),
cnames = c("a", "b", "c"), varnames = "y", idvar = 1)
xy <- DSC(x,y)
xyNorm <- normalize(xy)
tabresul(xyNorm)
## print("Show all elementary events")
xy1 <- addToDSM(normalize(DSC(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)

```

truncateDSM

*Truncation of the mass function of a DSM***Description**

When working with large frames of discernment, the DSM resulting of repeated application of Dempster's Rule of Combination can become quite large. One way to handle this situation is to group together subsets whose mass is less than a small threshold value. The function `DSMtrunc` is used for this purpose.

Usage

```
truncateDSM(x, seuil, use_ssnames = FALSE)
```

```
bcaTrunc(x, seuil, use_ssnames = FALSE)
```

Arguments

<code>x</code>	A DSM to truncate.
<code>seuil</code>	A treshold value
<code>use_ssnames</code>	Put TRUE to use ssnames parameteer instead of description matrix. Default = FALSE.

Value

tr_x The DSM object truncated.

Author(s)

Claude Boivin

Examples

```
x <- DSM(tt = matrix(c(0,1,0,0,
0,0,1,1,
1,1,0,0,
1,0,1,0,
0,1,1,0,
1,1,1,1),ncol=4, byrow = TRUE), m = c(0.2, 0.5, 0.06, 0.04, 0.03, 0.17),
cnames = c("a", "b", "c", "d"))
printDSM(x)
tr_x <- truncateDSM(x, seuil = 0.1)
printDSM(tr_x)
```

ttmatrix

Construct a description matrix from a list of subsets names.

Description

Construct a description matrix from a list of subsets names.

Usage

```
ttmatrix(x, sparse = "no")
```

Arguments

x A list of names
sparse = c("yes","no") whether to use sparse matrix. Default = "no".

Value

ttmat A corresponding logical description matrix

Author(s)

Claude Boivin

Examples

```
subsets_names <- list(c("b", "c"), "b", c("a", "b", "c"))
ttmatrix(subsets_names)
znames <- list("empty", "a", c("b", "c"), c("a", "b"), c("a", "b", "c") )
print(ttmatrix(znames) )
print(ttmatrix(znames, sparse = "yes") )
```

uproj

Extension of a relation

Description

This function works on a basic chance assignment (bca) **x** defined on a single variable or more. A relation of reference is given, and an extension of the space of **x** is made to the larger product space of the relation of reference. The basic chance assignment to extend and the relation of reference must have at least one common variable for the extension to occur.

Usage

```
uproj(rel1, relRef)

extmin(rel1, relRef)
```

Arguments

rel1	An object of class <code>DSMspec</code> , i.e. a basic chance assignment defined on 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 be a relation already existing or simply the the vacuous relation defined on the product set of variables of interest.

Value

the resulting extended bca.

Author(s)

Claude Boivin

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

```
# Making a vacuous reference relation and extending a DSM 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 <- jointDSM(tt = init_tt, spec = init_spec,
infovar = init_info,
varnames = c("Sail", "Loading", "Weather", "Repairs"),
relnb = 0)
# a DSMspec defined on one variable
l_rel <- DSM(tt = 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"))),
varnames = c("Loading"),
infovar = matrix(c(7,1), ncol = 2,
dimnames = list(NULL, c("relnb", "depth"))))
z <- uproj(l_rel, relRef)
prmatrix(t(z$tt), collab = rep("", nrow(z$tt)))
```

Index

- * datasets
 - ads, 4
 - captain_result, 5
 - dlfm, 7
 - fw, 20
 - mrf, 27
 - mrt, 28
 - swr, 40
- addTobca (*addToDSM*), 3
- addToDSM, 3
- ads, 4
- aplDecode (*decode*), 6
- aplEncode (*encode*), 18
- aplShape (*shape*), 38
- available. (*DSC*), 13
- bca (*DSM*), 15
- bcaPrint (*printDSM*), 34
- bcaRel (*jointDSM*), 23
- bcaTrunc (*truncateDSM*), 42
- belplau (*DSA*), 11
- belplauH (*DSAh*), 12
- bpa (*DSM*), 15
- calculations (*DSC*), 13
- captain_result, 5
- commonality, 5
- cores (*DSC*), 13
- decode, 6
- dlfm, 7
- DoSSNames, 8
- dotprod, 9
- dproj, 10
- DSA, 11
- DSAh, 12
- DSC, 13
- DSM, 3, 11, 14, 15
- dsrwon (*DSC*), 13
- elim (*dproj*), 10
- encode, 18
- extFrame (*extSSR*), 19
- extmin (*uproj*), 44
- extSSR, 19
- fw, 20
- inters, 21
- intersBySSName, 22
- jointDSM, 17, 23
- make (*DSC*), 13
- marrayToMatrix, 25
- matrixToMarray, 26
- mrf, 27
- mrt, 28
- multiple (*DSC*), 13
- nameCols, 29
- nameCols_prod, 29
- nameRows, 17, 24, 30
- normalize, 13, 31
- nzdsr (*normalize*), 31
- of (*DSC*), 13
- peeling, 32
- plautrans (*PT*), 36
- printDSM, 34
- productSpace, 35
- PT, 36
- PTsimDSM, 37
- shape, 38
- SSR, 39
- swr, 40
- tabresul, 41
- The (*DSC*), 13

truncatedSM, [42](#)

ttmatrix, [43](#)

uproj, [44](#)

use (*DSC*), [13](#)