

# Package ‘dst’

August 26, 2024

**Type** Package

**Title** Using the Theory of Belief Functions

**Encoding** UTF-8

**Version** 1.7.1

**Date** 2024-08-24

**Author** Claude Boivin <webapp.cb@gmail.com> Peiyuan Zhu <garyzhuc@gmail.com>

**Maintainer** Peiyuan Zhu <garyzhuc@gmail.com>

**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** 7.3.1

**Depends** R (>= 3.5.0)

**Imports** dplyr,  
ggplot2,  
tidyr,  
Matrix,  
methods,  
parallel,  
rlang,  
utils

**Suggests** igraph,  
knitr,  
rmarkdown,  
tidyverse,  
testthat

**VignetteBuilder** knitr

**R topics documented:**

addTobca . . . . .	3
ads . . . . .	4
bca . . . . .	5
bcaNorm . . . . .	7
bcaPrint . . . . .	8
bcaPrintLarge . . . . .	8
bcaRel . . . . .	9
bcaTrunc . . . . .	11
belplau . . . . .	12
belplauEval . . . . .	13
belplauH . . . . .	14
belplauHLogsumexp . . . . .	15
belplauHQQ . . . . .	16
belplauLogsumexp . . . . .	17
belplauPlot . . . . .	18
captain_result . . . . .	20
commonality . . . . .	20
decode . . . . .	21
dlfm . . . . .	22
DoSSnames . . . . .	23
dotprod . . . . .	23
doubles . . . . .	24
dsrwon . . . . .	25
dsrwonLogsumexp . . . . .	27
elim . . . . .	28
encode . . . . .	29
extFrame . . . . .	30
extmin . . . . .	31
fw . . . . .	32
inters . . . . .	33
intersBySSName . . . . .	34
logsum . . . . .	35
marrayToMatrix . . . . .	35
matrixToMarray . . . . .	36
mFromMarginal . . . . .	37
mFromQQ . . . . .	37
mobiusInvHQQ . . . . .	38
mrf . . . . .	39
mrt . . . . .	40
nameCols . . . . .	41
nameCols_prod . . . . .	41
nameRows . . . . .	42
nzdsr . . . . .	43
nzdsrLogsumexp . . . . .	44
peeling . . . . .	45
plautrans . . . . .	47
productSpace . . . . .	48
reduction . . . . .	49
shape . . . . .	50
swr . . . . .	50

tabresul . . . . .	51
ttmatrix . . . . .	52
ttmatrixFromMarginal . . . . .	53
ttmatrixFromQQ . . . . .	54
ttmatrixPartition . . . . .	55

Index

56

---

addTobca	<i>Add some elements of 0 mass to an existing basic chance assignment.</i>
----------	--

---

## Description

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

## Usage

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

## Arguments

x	A basic chance assignment (see <a href="#">bca</a> ).
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 <a href="#">bca</a> ). Each row of the matrix identify a subset of the set of possible values.
f	Deprecated. Old name for tt matrix.

## Value

x The original basic chance assignment x augmented with the added subsets defined by tt.

## Author(s)

Claude Boivin

## Examples

```
y <- 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"), idvar = 1)
addTobca(y, matrix(c(0,1,0,0,0,1, 0,1,1), nrow = 3, byrow = TRUE))
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"), idvar = 1)
xy <- dsrwon(x,y)
xy1 <- addTobca(nzdsr(xy), matrix(c(0,1,0,0,0,1), nrow = 2, byrow = TRUE))
xy1
# add all singletons to a bca
addTobca(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  $t t$  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  $t t$  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.

bca

*Basic chance assignment mass function***Description**

Function `bca` is used to define subsets of a finite set  $\Theta$  of possible values and to assign their corresponding mass value.

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

**Usage**

```
bca(
  tt = NULL,
  m,
  qq = NULL,
  cnames = NULL,
  con = NULL,
  ssnames = NULL,
  idvar = NULL,
  infovar = NULL,
  varnames = NULL,
  valuenames = NULL,
  inforel = NULL
)
```

**Arguments**

<code>tt</code>	Mandatory. A (0,1)-matrix or a boolean matrix. The number of columns must match the number of elements (values) of the frame of discernment $\Theta$ . Each row is a subset of $\Theta$ . The last row is the frame $\Theta$ , represented by a vector of 1's.
<code>m</code>	A numeric vector of length equal to the number of rows of the matrix <code>tt</code> . Values of <code>m</code> must lie in the interval $(0, 1]$ and must add to one. The mass $m(k)$ represents the chance value allotted to the proposition represented by the row <code>k</code> of the matrix <code>tt</code> .
<code>qq</code>	Commonality functions from the frame of discernment to $[0, 1]$
<code>cnames</code>	A character vector containing the names of the elements of the frame of discernment $\Theta$ . The length must be equal to the number of elements of $\Theta$ . The names are first searched in the <code>valuenames</code> parameter. If <code>NULL</code> , column names of the matrix <code>tt</code> are taken if present. Otherwise, names are generated.
<code>con</code>	The measure of conflict can be provided. 0 by default.
<code>ssnames</code>	A list of subsets names which will be obtained from the column names of the <code>tt</code> matrix.
<code>idvar</code>	The number given to the variable. A number is necessary to manage relations between variables and make computations on a graph. 0 if omitted.
<code>infovar</code>	A two-column matrix containing variable identification numbers and the number of elements of the variable. Generated if omitted.

varnames	The name of the variable. Generated if omitted.
valuenames	A list of the names of the variables with the name of the elements of their frame of discernment.
inforel	Not used here. Defined within function <a href="#">bcaRel</a> .

## Details

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

## Value

y An object of class `bcaspec` called a bca for "basic chance assignment":

- `tt` The table of focal elements. Rownames of the matrix of focal elements are generated from the column names of the elements of the frame. See [nameRows](#) for details.
- `qq` Commonality functions from the frame of discernment to  $[0, 1]$
- `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 frame of discernment.
- `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 frame of discernment (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 [bcaRel](#).

## 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

## Examples

```
tt<- t(matrix(c(1,0,1,1),ncol = 2))
m<- c(.9,.1)
cnames <- c("yes","no")
bca(tt, m)
bca(tt, m, cnames)
tt1<- t(matrix(c(1,0,1,1),ncol = 2))
colnames(tt1) <- c("yes", "no")
m <- c(.9, .1)
bca(tt=tt1, m, idvar = 1)
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"), idvar = 1)
y <- 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 = "y", idvar = 1)
vacuous <- bca(matrix(c(1,1,1), nrow = 1), m = 1, cnames = c("a","b","c"), ssnames = c("a","b","c"))
```

bcaNorm

*Computer norm between two basic chance assignment objects*

## Description

Computer norm between two basic chance assignment objects

## Usage

```
bcaNorm(x, y, p = 1)
```

## Arguments

x	A bca to evaluate norm.
y	A bca to evaluate norm.
p	exponent parameter of the norm

## Value

a number of norm evaluation

## 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)
y1y2<-dsrwon(y1,y2)
bcaNorm(y1y2,y1)
```

---

bcaPrint	<i>Simple printing of the tt matrix and mass values of a basic chance assignment (bca)</i>
----------	--

---

### Description

This utility function does a simple printing of a bca

### Usage

```
bcaPrint(x)
```

### Arguments

x	A list of class bcaspec.
---	--------------------------

### Value

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

### Author(s)

Claude Boivin

### Examples

```
z <- 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"), idvar = 1)
bcaPrint(z)
```

---

bcaPrintLarge	<i>Print summary statistics of large mass functions</i>
---------------	---

---

### Description

Print summary statistics of large mass functions

### Usage

```
bcaPrintLarge(
  x,
  info_list = "all",
  num_top_mass = 10,
  cut_width_size = 10,
  cut_width_m = 1e-05
)
```



**Arguments**

<code>x</code>	A basic chance assignment (see <a href="#">bca</a> ).
<code>info_list</code>	="all" statistics to be printed in a vector of characters <ul style="list-style-type: none"> <li>• "all": everything</li> <li>• "basic_subset_stat": basic statistics of subsets</li> <li>• "comp_subset_stat": more comprehensive statistics of subsets</li> <li>• "basic_mass_stat": basic statistics of masses</li> <li>• "comp_mass_stat": more comprehensive statistics of masses</li> <li>• "basic_joint_stat": basic statistics of masses vs subsets</li> <li>• "comp_joint_stat": more comprehensive statistics of masses vs subsets</li> </ul>
<code>num_top_mass</code>	= 10 number of top masses to be printed
<code>cut_width_size</code>	width of a cut among subset sizes
<code>cut_width_m</code>	width of a cut among masses

**Value**

- table of basic and more comprehensive statistics of subsets
- table of basic and more comprehensive statistics of masses
- table of basic and more comprehensive statistics of masses vs subsets

**Author(s)**

Peiyuan Zhu

**Examples**

```

if (requireNamespace("tidyverse", quietly = TRUE) ) {
  ## library(tidyverse)
  x <- bca(tt = matrix(c(1,1,0,1,1,1), nrow = 2, byrow = TRUE), m = c(0.8, 0.2), cnames = c(1,2,3))
  bcaPrintLarge(x)
}

```

---

bcaRel

---

*Representation of a mass function in a product space*


---

**Description**

This function is used to represent a relation between two or more variables in their product space  $P$ . The relation can be described by more than one subset of  $P$ . Each subset can also include more than one element. Complete disjunctive coding is used to represent one element in the input matrix of the function.

**Usage**

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

**Arguments**

tt	The description matrix of the subsets establishing the relation. This matrix is obtained by putting the variables side by side, as in a truth table representation. For each variable, there are as many columns as possible values. Each row of the matrix is an element of a subset. Each element is described by a sequence of 0 (absence of value of a variable) or 1 (presence of value). This forms a complete disjunctive coding. CAUTION: Variables 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 sub-assemblies. Second column: the mass values of the sub-assemblies. If the subset has more than one element, the number of the subset and its associated mass value are repeated to match the number of elements in 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.
varnames	The names of the variables.
valuenames	A list of the names of the variables with the name of the elements of their frame of discernment.
relnb	A number given to the relation. Set at 0 if omitted.
infovarnames	Deprecated. Old name for varnames.
infovaluenames	Deprecated. Old name for valuenames.

**Value**

An object of class `bcaspec` called a `bca` for "basic chance assignment". This is a list containing the following components:

- `con` The measure of conflict.
- `tt` The resulting table of subsets. Rownames of the matrix of subsets are generated from the column names of the elements of the product frame. See [nameRows](#) for details.
- `spec` The resulting two-column matrix of specification numbers with associated mass values.
- `infovar` The two-column matrix of variables number and size given in the input data.
- `valuenames` A list of the names of the variables with the name of the elements of their frame of discernment.
- `inforel` A two-column matrix containing the relation number and the depth (number of variables) of the relation.

**Author(s)**

Claude Boivin

**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.

# 1. The tt table of the logical implication
ttrwf <- matrix(c(0,1,1,0,1,0,1,0,1,0,0,1,1,1,1,1),
  nrow = 4, byrow = TRUE,
  dimnames = list(NULL, c("rWdy", "rWdn", "Ry", "Rn"))) )

# 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")))

# 3. 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,
  varnames = c("RdWorks", "Rain"), relnb = 6)
```

bcaTrunc

*Truncation of a basic chance assignment mass function***Description**

When working with large frames of discernment, the bca resulting of repeated application of Dempster's Rule of Combination can become big. One way to handle this situation could be to group subsets whose mass is less than a small threshold value. The function `bcaTrunc` serves this purpose to reduce a large bca to its main elements.

**Usage**

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

**Arguments**

<code>x</code>	A bca to truncate.
<code>seuil</code>	A threshold value
<code>use_ssnames</code>	Put TRUE to use <code>ssnames</code> parameter instead of description matrix. Default = FALSE.

**Value**

`tr_x` The bca object truncated.

**Author(s)**

Claude Boivin

**Examples**

```
x <- bca(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"))
bcaPrint(x)
tr_x <- bcaTrunc(x, seuil = 0.1)
bcaPrint(tr_x)
```

belplau

*Calculation of the degrees of Belief and Plausibility of a basic chance assignment (bca).*

**Description**

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

**Usage**

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

**Arguments**

x                    A basic chance assignment mass function (see [bca](#)).  
remove                = TRUE: Exclude subsets with zero mass.  
h                     = NULL: Hypothesis to be tested. Description matrix in the same format than x\$tt

**Details**

The degree of belief Bel is defined by:

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

for every subset B of A.

The degree of plausibility pl is defined by:

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

for every subset B of the frame of discernment.

The plausibility ratio of a focal element A versus its contrary not A is defined by:  $Pl(A)/(1 - Bel(A))$ .

**Value**

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

- Column 1: the degree of Belief bel;
- Column 2: the degree of Disbelief (belief in favor of the contrary hypothesis) disbel;
- Column 3: the degree of Epistemic uncertainty unc;
- Column 4: the degree of 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.

**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)
belplau(x)
y <- 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 = "y", idvar = 1)
xy <- nzdsr(dsrwon(x,y))
belplau(xy)
print("compare all elementary events")
xy1 <- addTobca(x = xy, tt = matrix(c(0,1,0,0,0,1), nrow = 2, byrow = TRUE))
belplau(xy1)
belplau(xy1, remove = TRUE)
belplau(xy1, h = matrix(c(1,0,0,0,1,1), nrow = 2, byrow = TRUE))
```

---

belplauEval

---

*Evaluate A, B errors*


---

**Description**

Calculate error A, B, and total error A+B by comparing two vectors as defined below. One vector represents the truth and the other represents a numerical quantity of importance.

- Error A: out of all the comparisons between two elements, what proportion of errors are due to indicating an irrelevant element as more important than a relevant element
- Error B: out of all the comparisons between two elements, what proportion of errors are due to indicating a relevant element as less important than an irrelevant element
- Total error A+B: the sum of quantity A and quantity B

**Usage**

```
belplauEval(
  belplau_mat,
  true_order,
  var = "rplau",
  err = "A",
  is_belplau = TRUE
)
```

**Arguments**

belplau_mat	belplau matrix e.g. belplau(bca) or a numerical vector quantifying order of importance of the elements of the frame.
true_order	a binary vector representing the truth. 1 means relevant and 0 means not relevant.
var	= "rplau" column name of the belplau matrix to be used as ordering.
err	= "A" kind of error to be evaluated. Can also take value "B" or "A+B".
is_belplau	= TRUE whether bel_plau is indeed a belplau matrix or just a numerical vector quantifying order of importance of elements.

**Value**

A number in  $[0, 1]$  of error A, B, or total error A+B.

**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)
belplau(x)
y <- 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 = "y", idvar = 1)
xy <- nzdsr(dsrwon(x,y))
z<-belplau(xy,h=ttmatrixPartition(xy$infovar[2],xy$infovar[2]))
belplauEval(z,c(0,1,0))
```

---

belplauH

*Calculate belief, disbelief, unknown, plausibility, plausibility ratio*

---

**Description**

Calculate belief, disbelief, unknown, plausibility, plausibility ratio

**Usage**

```
belplauH(MACC, W2, h)
```

**Arguments**

MACC	Vector of masses e.g. x\$spec[,2]
W2	Description matrix e.g. x\$tt
h	H hypotheses to be tested, same format as x\$tt

**Value**

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

- Column 1: the degree of Belief bel;
- Column 2: the degree of Disbelief (belief in favor of the contrary hypothesis) disbel;
- Column 3: the degree of Epistemic uncertainty unc;
- Column 4: the degree of Plausibility plau;
- Column 5: the Plausibility ratio rplau.

**Author(s)**

Peiyuan Zhu

**Examples**

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

---

belplauHLogsumexp	<i>Calculate belief, disbelief, unknown, plausibility, plausibility ratio with logsumexp</i>
-------------------	--

---

**Description**

Calculate belief, disbelief, unknown, plausibility, plausibility ratio with logsumexp

**Usage**

```
belplauHLogsumexp(MACC, W2, h)
```

**Arguments**

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

**Value**

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

- Column 1: the degree of Belief bel;
- Column 2: the degree of Disbelief (belief in favor of the contrary hypothesis) disbel;
- Column 3: the degree of Epistemic uncertainty unc;
- Column 4: the degree of Plausibility plau;
- Column 5: the Plausibility ratio rplau.

**Author(s)**

Peiyuan Zhu

**Examples**

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

---

belplauHQQ	<i>Compute belief, disbelief, unknown, plausibility, plausibility ratio based on commonality function</i>
------------	---

---

**Description**

Compute belief, disbelief, unknown, plausibility, plausibility ratio based on commonality function

**Usage**

```
belplauHQQ(qq, h = NULL)
```

**Arguments**

qq	Commonality function
h	= NULL Hypothesis to be evaluated

**Value**

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

- Column 1: the degree of Belief bel;
- Column 2: the degree of Disbelief (belief in favor of the contrary hypothesis) disbel;
- Column 3: the degree of Epistemic uncertainty unc;
- Column 4: the degree of Plausibility plau;
- Column 5: the Plausibility ratio rplau.



**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])
belplauHQQ(qq,h=matrix(c(0,1,0), nrow=1, byrow=TRUE))
```

belplauLogsumexp

*Calculation of the degrees of Belief and Plausibility of a basic chance assignment (bca) with logsumexp.*

**Description**

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

**Usage**

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

**Arguments**

x                      A basic chance assignment mass function (see [bca](#)).

remove                = TRUE: Exclude subsets with zero mass.

h                      = NULL: Hypothesis to be tested. Description matrix in the same format than x\$tt

**Details**

The degree of belief Bel is defined by:

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

for every subset B of A.

The degree of plausibility pl is defined by:

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

for every subset B of the frame of discernment.

The plausibility ratio of a focal element A versus its contrary not A is defined by:  $Pl(A)/(1 - Bel(A))$ .

**Value**

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

- Column 1: the degree of Belief bel;
- Column 2: the degree of Disbelief (belief in favor of the contrary hypothesis) disbel;
- Column 3: the degree of Epistemic uncertainty unc;
- Column 4: the degree of 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.

**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)
belplau(x)
y <- 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 = "y", idvar = 1)
xy <- nzdsr(dsrwon(x,y))
belplau(xy)
print("compare all elementary events")
xy1 <- addTobca(x = xy, tt = matrix(c(0,1,0,0,0,1), nrow = 2, byrow = TRUE))
belplau(xy1)
belplau(xy1, remove = TRUE)
belplau(xy1, h = matrix(c(1,0,0,0,1,1), nrow = 2, byrow = TRUE))
```

---

belplauPlot

*Plot belplau matrix*

---

**Description**

Plot belplau matrix

**Usage**

```
belplauPlot(
  belplau_mat,
  xlab,
  color,
  y = "rplau",
  x = "index",
  levels = NULL,
  legend_title = "",
  main_title = "",
  is_log_scale = TRUE,
  is_negative = FALSE,
  is_factor = FALSE
)
```

**Arguments**

belplau_mat	Belplau matrix e.g. belplau(bpa) or a numerical vector quantifying order of importance of the elements of the frame
xlab	X-axis labels e.g. c("1:34", "35:68", "69:101")
color	Color of xlab e.g. c(0,1,0)
y	= "rplau": column name of belplau matrix. Ignore if it's not belplau matrix.
x	= "index": x-axis name
levels	= NULL: levels of color in order
legend_title	= "": title of legend
main_title	= "": main title
is_log_scale	= TRUE Whether to use log-scale
is_negative	= TRUE Whether to multiple by -1
is_factor	= FALSE Whether to plot all x labels

**Value**

a plot of a column of the belplau matrix or a numerical vector quantifying order of importance of the elements of the frame

**Author(s)**

Peiyuan Zhu

**Examples**

```
bpa <- 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)
bel_plau <- belplau(bpa)
belplauPlot(bel_plau, c("a","b","c"), c(1,3,2))
```

---

captain_result	<i>The Captain's Problem. swr: 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)
```

**Arguments**

<code>tt</code>	Mass assignment set matrix
<code>m</code>	Mass assignment

**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])
qq(c(1,0,0))
```

---

 decode

---

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


---

**Description**

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

**Usage**

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

**Arguments**

<code>base</code>	A scalar or a numeric vector which describes the number system in which the data is coded.
<code>ind</code>	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	<i>Construct subsets names from column names of a tt matrix</i>
-----------	---

---

**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	<i>Generalized inner product of two matrices</i>
---------	--

---

**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", "Garipey", "Nellie", "Arcand"),
ncol = 2, byrow = TRUE)
colnames(team_names) <- c("First_name", "Last_name")
print("Where in the list are the person with first name Robert and where are the Doles?")
BobandDoles <- matrix(c("Robert", "", "", "Dole"),
ncol = 2, byrow = TRUE)
dotprod(team_names, t(BobandDoles),g="|",f=="") * (1:nrow(team_names))
```

---

doubles

---

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


---

**Description**

Recursive function.

**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

**Examples**

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

---

dsrwon

---

*Combination of two mass functions*


---

**Description**

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

**Usage**

```
dsrwon(
  x,
  y,
  mcores = "no",
  use_ssames = FALSE,
  use_qq = FALSE,
  varnames = NULL,
  relnb = NULL,
  skpt_tt = FALSE,
  infovarnames
)
```

**Arguments**

<code>x</code>	A basic chance assignment (see <a href="#">bca</a> ).
<code>y</code>	A basic chance assignment (see <a href="#">bca</a> ).
<code>mcores</code>	Make use of multiple cores ("yes") or not ("no"). Default = "no".
<code>use_ssames</code>	= TRUE to use ssames instead of tt matrix to do the intersections. Default = FALSE

<code>use_qq</code>	= TRUE to use qq instead of tt matrix to do the intersections. Default = FALSE
<code>varnames</code>	A character string to name the resulting variable. named "z" if omitted.
<code>relnb</code>	Identification number of the relation. Can be omitted.
<code>skpt_tt</code>	Skip reconstruction of tt matrix. Default = FALSE.
<code>infovarnames</code>	Deprecated. Old name for varnames.

## Details

The calculations make use of multiple cores available.

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

## Value

A basic chance assignment 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, pp. 57-61: Dempster's rule of combination.

## 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)
dsrwon(y1,y2)
# Sparse matrices
y1s <- y1
y2s <- y2
y1s$tt <- methods::as(y1$tt, "RsparseMatrix")
y2s$tt <- methods::as(y2$tt, "RsparseMatrix")
y1y2s <- dsrwon(y1s, y2s, use_ssnames = TRUE)

# using commonalities
bma <- bca(tt=matrix(c(1,1,0,1,rep(1,4)), ncol = 4, byrow = TRUE),
m = c(0.1, 0.9), cnames = c("a", "b", "c", "d"))
bma2 <- dsrwon(bma, bma, use_qq = TRUE)

vacuous <- bca(matrix(c(1,1,1), nrow = 1), m = 1, cnames = c("a","b","c"))
dsrwon(vacuous, vacuous)

```

dsrwonLogsumexp

*Combination of two mass functions with logsumexp***Description**

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

**Usage**

```
dsrwonLogsumexp(
  x,
  y,
  mcores = "no",
  use_ssnames = FALSE,
  use_qq = FALSE,
  varnames = NULL,
  relnb = NULL,
  skpt_tt = FALSE,
  infovarnames
)
```

**Arguments**

<code>x</code>	A basic chance assignment (see <a href="#">bca</a> ).
<code>y</code>	A basic chance assignment (see <a href="#">bca</a> ).
<code>mcores</code>	Make use of multiple cores ("yes") or not ("no"). Default = "no".
<code>use_ssnames</code>	= TRUE to use ssnames instead of tt matrix to do the intersections. Default = FALSE
<code>use_qq</code>	= TRUE to use qq instead of tt matrix to do the intersections. Default = FALSE
<code>varnames</code>	A character string to name the resulting variable. named "z" if omitted.
<code>relnb</code>	Identification number of the relation. Can be omitted.
<code>skpt_tt</code>	Skip reconstruction of tt matrix. Default = FALSE.
<code>infovarnames</code>	Deprecated. Old name for varnames.

**Details**

The calculations make use of multiple cores available.

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

**Value**

A basic chance assignment 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, pp. 57-61: Dempster's rule of combination.

**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)
dsrwonLogsumexp(y1,y2)
# Sparse matrices
y1s <- y1
y2s <- y2
y1s$tt <- methods::as(y1$tt, "RsparseMatrix")
y2s$tt <- methods::as(y2$tt, "RsparseMatrix")
y1y2s <- dsrwonLogsumexp(y1s, y2s, use_ssnames = TRUE)
vacuous <- bca(matrix(c(1,1,1), nrow = 1), m = 1, cnames = c("a","b","c"))
dsrwonLogsumexp(vacuous, vacuous)
```

---

elim

---

*Reduction of a relation*


---

**Description**

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

**Usage**

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

**Arguments**

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

**Value**

r The reduced relation

**Author(s)**

Claude Boivin

## Examples

```
# We construct a relation 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) <- "bcaspec"
bcaPrint(elim(wr_rel, xnb = 5))
bcaPrint(elim(wr_rel, xnb = 4))
```

---

encode

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

---

## Description

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

## Usage

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

## Arguments

<code>base</code>	A numeric vector which describes the number system in which we want to re-code the data.
<code>ind</code>	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.
```

---

 extFrame

*Extension of the frame of discernment of a variable*


---

**Description**

This function works on a basic chance assignment (bca)  $x$  defined on a single variable. It allows the addition of new values to the frame of discernment.

**Usage**

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

**Arguments**

<code>x</code>	An object of bca class, i.e. a basic chance assignment defined on one variable
<code>use_ssnames</code>	Default= FALSE. Put TRUE if use of subset names is wanted.
<code>lab</code>	A character vector containing the names of the elements to add to the frame of discernment.

**Value**

`zxtnd` The bca with its frame extended

**Author(s)**

Claude Boivin

**Examples**

```
s1_e1 <- bca(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 <- extFrame(s1_e1, lab = "S3", use_ssnames = TRUE)
s13 <- extFrame(s1_e1, lab = "S3")
```

---

extmin	<i>Extension of a relation</i>
--------	--------------------------------

---

### 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

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

### Arguments

rel1	An object of class <code>bcaspec</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 bca to its space.
init_tt = matrix(rep(1,10),nrow = 1,
dimnames = list(NULL, c("3", "2", "1", "0",
"true", "false", "foul", "fair", "true", "false"))) )
init_spec <- matrix(c(1,1), ncol = 2,
dimnames = list(NULL, c("specnb", "mass")))
init_info <- matrix(c(3,4,7,8,4,2,2,2), ncol = 2,
dimnames = list(NULL, c("varnb", "size"))) )
relRef <- bcaRel(tt = init_tt, spec = init_spec,
```

```

infovar = init_info,
varnames = c("Sail", "Loading", "Weather", "Repairs"),
relnb = 0)
# a bcaspec defined on one variable
l_rel <- bca(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 <- extmin(l_rel, relRef)
prmatrix(t(z$tt), collab = rep("", nrow(z$tt)))

```

fw

*The Captain's Problem. fw: Relation between variables Forecast of the weather (F) and Weather at sea (W)*

## 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**

<code>x</code>	A (0,1)-matrix or a boolean matrix of M rows by K columns, or a vector of length K.
<code>y</code>	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]])

```

---

logsum	<i>Adding small probabilities</i>
--------	-----------------------------------

---

**Description**

Adding small probabilities

**Usage**

```
logsum(l1, l2)
```

**Arguments**

l1	log probabilities
l2	log probabilities

**Value**

sum of probabilities  $\exp(l1) + \exp(l2)$

**Author(s)**

Peiyuan Zhu

**Examples**

```
# sum of two 1e-5
exp(logsum(log(1e-5), log(1e-5)))
```

---

marrayToMatrix	<i>Transformation of an array data to its matrix representation</i>
----------------	---

---

**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 <i>tt</i> 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

*Transformation of the tt matrix of a relation***Description**

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

**Usage**

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

**Arguments**

<code>tt</code>	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.
<code>valuenames</code>	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 ) )
```

---

mFromMarginal	<i>Construct m vector of a bca from marginal probabilities</i>
---------------	--

---

**Description**

Construct m vector of a bca from marginal probabilities

**Usage**

```
mFromMarginal(
  marg_probs,
  a = 1e-05,
  simple = FALSE,
  min_prob = 0,
  max_prob = 2
)
```

**Arguments**

marg_probs	vector of marginal probabilities
a	=1e-5 probability that the sample is reliable
simple	=TRUE whether to use simple support function
min_prob	=0 lower bound on marginal probabilities
max_prob	=2 upper bound on marginal probabilities

**Value**

vector of probability masses obtained from uniformly sampling the cut

**Author(s)**

Peiyuan Zhu

**Examples**

```
x <- c(2,2,1.5,1.2,1,0,0)
mFromMarginal(x, simple=FALSE)
```

---

mFromQQ	<i>Construct a mass vector from qq function.</i>
---------	--

---

**Description**

Construct a mass vector from qq function.

**Usage**

```
mFromQQ(qq, tt)
```

**Arguments**

qq                      Commonality function  
 tt                      logical description matrix from ttmatrixFromQQ

**Value**

m A corresponding mass vector

**Author(s)**

Peiyuan Zhu

**Examples**

```
tt<- t(matrix(c(1,0,1,1),ncol = 2))
m<- c(.9,.1)
cnames <- c("yes","no")
x<- bca(tt, m, cnames=cnames)
mFromQQ(x$qq, x$tt)
```

---

mobiusInvHQQ

*Mobius inversion of commonality function*


---

**Description**

Mobius inversion of commonality function

**Usage**

```
mobiusInvHQQ(qq, h)
```

**Arguments**

qq                      Commonality function  
 h                      Hypothesis to be evaluated

**Value**

m Mass of the hypothesis

**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])
mobiusInvHQQ(qq, matrix(c(0,1,0,1,1,0), nrow = 2, byrow = TRUE))
```

mrf

*The Captain's Problem. mrf: Relation between variables No Maintenance ( $M = \text{false}$ ) 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 = 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  $t t$  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  $t t$  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 tt 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 tt 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**

<code>tt</code>	A (0,1)-matrix or a boolean matrix.
<code>f</code>	Deprecated. Old name for <code>tt</code> 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), ncol = 3, byrow = TRUE)
colnames(tt) <- c("A", "B", "C")
rownames(tt) <- nameRows(tt)
tt
```

nzdsr

*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 `nzdsr` normalizes the result of function `dsrwon` by dividing the mass value of the non-empty subsets by 1 minus the mass of the empty set.

**Usage**

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

**Arguments**

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

```

---

nzdsrLogsumexp

*Normalization of a basic chance assignment with logsumexp*


---

## 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 `nzdsr` normalizes the result of function `dsrwon` by dividing the mass value of the non-empty subsets by 1 minus the mass of the empty set.

## Usage

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

## Arguments

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

```

---

plautrans

*Plausibility transformation of the singletons of a frame*


---

### Description

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

### Usage

```
plautrans(x)
```

### Arguments

**x**                      A bca mass function.

### Details

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

### Value

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

### Author(s)

Claude Boivin

### 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(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)
plautrans(x)
```

---

productSpace	<i>Product space representation of a relation</i>
--------------	---

---

## 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

```
tffw <- matrix(c(1,0,1,0,0,1,0,1,1,1,1),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(tffw) <- nameRows(tffw)
tffw
productSpace(tt = tffw, specnb = specfw, infovar = infovarfw)
```



---

reduction	<i>Summary of a vector for any operator.</i>
-----------	--

---

## Description

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

## Usage

```
reduction(x, f)
```

## Arguments

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

## Value

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

## Author(s)

Claude Boivin

## References

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

## Examples

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

---

shape	<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"))
```

---

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.

---

tabresul

---

*Prepare a table of results*


---

**Description**

This utility function is a more detailed version of the `belpau` 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 `belpau`, 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 `bca` before calling function `tabresul`.

**Usage**

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

**Arguments**

`x`                      A basic chance assignment (bca)

`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 <- 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)
y <- 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 = "y", idvar = 1)
xy <- dsrwon(x,y)
xyNorm <- nzdsr(xy)
tabresul(xyNorm)
## print("Show all elementary events")
xy1 <- addTobca(nzdsr(dsrwon(x,y)),
matrix(c(0,1,0,0,0,1),
nrow = 2, byrow = TRUE))
tabresul(xy1)
## print("Remove focal elements with 0 mass")
tabresul(xy1, removeZeroes = TRUE)
print("Retain singletons only")
tabresul(xy1, singletonsOnly = TRUE)
```

---

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") )
```

---

ttmatrixFromMarginal    *Construct tt matrix of a bca from marginal probabilities*

---

**Description**

Construct tt matrix of a bca from marginal probabilities

**Usage**

```
ttmatrixFromMarginal(
  marg_probs,
  from_above = FALSE,
  simple = FALSE,
  min_prob = 0,
  max_prob = 2
)
```

**Arguments**

**marg\_probs**            marginal probabilities

**from\_above**            =TRUE whether to cut marginal probabilities from above

**simple**                 =TRUE whether to use simple support function

**min\_prob**              =0 lower bound on marginal probabilities

**max\_prob**              =2 upper bound on marginal probabilities

**Value**

matrix of possible subsets obtained from the cuts

**Author(s)**

Peiyuan Zhu

**Examples**

```
x <- c(2,2,1.5,1.2,1,0,0)
ttmatrixFromMarginal(x, FALSE)
```

---

ttmatrixFromQQ

---

*Construct a description matrix from qq function.*


---

**Description**

Construct a description matrix from qq function.

**Usage**

```
ttmatrixFromQQ(qq, n, valuenames)
```

**Arguments**

qq	Commonality function
n	Dimension of the frame
valuenames	Vector of valuenames

**Value**

ttmat A corresponding logical description matrix

**Author(s)**

Peiyuan Zhu

**Examples**

```
tt<- t(matrix(c(1,0,1,1),ncol = 2))
m<- c(.9,.1)
cnames <- c("yes","no")
x<- bca(tt, m, cnames=cnames)
ttmatrixFromQQ(x$qq,as.integer(x$inforvar[1,2]), unlist(x$valuenames))
```

---

ttmatrixPartition	Create partition matrix
-------------------	-------------------------

---

**Description**

Create partition matrix

**Usage**

```
ttmatrixPartition(n, m)
```

**Arguments**

n	partition size
m	size of the set to be partitioned

**Value**

h binary partition matrix of size n by m

**Author(s)**

Peiyuan Zhu

**Examples**

```
# test singleton hypotheses
x <- bca(tt = matrix(c(1,1,0,1,1,1), nrow = 2, byrow = TRUE), m = c(0.8, 0.2), cnames = c(1,2,3))
pa <- ttmatrixPartition(x$infovar[2], x$infovar[2])
belplau(x, h=pa)
```

# Index

## \* datasets

- ads, [4](#)
- captain\_result, [20](#)
- dlfm, [22](#)
- fw, [32](#)
- mrf, [39](#)
- mrt, [40](#)
- swr, [50](#)

- addTobca, [3](#)
- ads, [4](#)
- aplDecode (decode), [21](#)
- aplEncode (encode), [29](#)
- aplRDV (reduction), [49](#)
- aplShape (shape), [50](#)

- bca, [3](#), [5](#), [9](#), [12](#), [17](#), [25](#), [27](#)
- bcaNorm, [7](#)
- bcaPrint, [8](#)
- bcaPrintLarge, [8](#)
- bcaRel, [6](#), [9](#)
- bcaTrunc, [11](#)
- belplau, [12](#)
- belplauEval, [13](#)
- belplauH, [14](#)
- belplauHLogsumexp, [15](#)
- belplauHQQ, [16](#)
- belplauLogsumexp, [17](#)
- belplauPlot, [18](#)
- bpa (bca), [5](#)

- captain\_result, [20](#)
- commonality, [20](#)

- decode, [21](#)
- dlfm, [22](#)
- DoSSNames, [23](#)
- dotprod, [23](#)
- doubles, [24](#)
- dsrwon, [25](#)
- dsrwonLogsumexp, [27](#)

- elim, [28](#)
- encode, [29](#)
- extFrame, [30](#)

- extmin, [31](#)

- fw, [32](#)

- inters, [33](#)
- intersBySSName, [34](#)

- logsum, [35](#)

- marrayToMatrix, [35](#)
- matrixToMarray, [36](#)
- mFromMarginal, [37](#)
- mFromQQ, [37](#)
- mobiusInvHQQ, [38](#)
- mrf, [39](#)
- mrt, [40](#)

- nameCols, [41](#)
- nameCols\_prod, [41](#)
- nameRows, [6](#), [10](#), [42](#)
- nzdsr, [25](#), [27](#), [43](#)
- nzdsrLogsumexp, [44](#)

- peeling, [45](#)
- plautrans, [47](#)
- productSpace, [48](#)

- reduction, [49](#)

- shape, [50](#)
- swr, [50](#)

- tabresul, [51](#)
- ttmatrix, [52](#)
- ttmatrixFromMarginal, [53](#)
- ttmatrixFromQQ, [54](#)
- ttmatrixPartition, [55](#)