

# Package ‘nCopula’

August 2, 2017

**Type** Package

**Title** Copula Construction Tools

**Version** 0.1.0

**Description**

Construct and use hierarchical Archimedean copulas with multivariate compound distributions

**Depends** R (>= 3.3), copula

**Imports** Deriv, gtools, mgcv, stringr, stringi, compiler, gsl, Ryacas,  
methods, Matrix, stabledist, beeper

**License** GPL (>= 2)

**LazyData** FALSE

**RoxygenNote** 6.0.1

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**Author** Simon-Pierre Gadoury [aut, cre]

**Maintainer** Simon-Pierre Gadoury <spgadou@me.com>

## R topics documented:

addNode . . . . .	2
AMH . . . . .	3
Clayton . . . . .	3
Frank . . . . .	4
GAMMA . . . . .	5
GeneticCodes . . . . .	5
GEO . . . . .	6
Gumbel . . . . .	7
InvLap . . . . .	7
Lap . . . . .	8
LOG . . . . .	9
Node . . . . .	10
pCompCop . . . . .	11
pCop . . . . .	11
rCompCop . . . . .	12
rCop . . . . .	13
rStruc . . . . .	14
<b>Index</b>	<b>15</b>

---

addNode	<i>Add a Distribution</i>
---------	---------------------------

---

**Description**

The function addNode is used to create and add a class of type "Mother" and/or "Child" in the global environment.

**Usage**

```
addNode(type, pp, name_short, name_long, Laplace, LaplaceInv, PGF = NULL,
        PGFInv = NULL, simul, cop = NULL, cop_name = "The copula")
```

**Arguments**

type	the type of the node (either 'Mother', 'Child' or 'Both')
pp	the parameter of the distribution used in the character strings
name_short	the short name of the distribution (ex.: 'log' for the logarithmic distribution)
name_long	the long name of the distribution (ex.: 'logarithmic' for the logarithmic distribution)
Laplace	the LST of the distribution (character), where
LaplaceInv	the inverse LST of the distribution (character)
PGF	the pgf of the distribution (character), if type is 'Mother' or 'Both'
PGFInv	the inverse pgf of the distribution (character), if type is 'Mother' or 'Both'
simul	function to sample from the distribution
cop	function to create a corresponding Archimedean copula class (ex.: for a GEO, it is an AMH copula), can be NULL
cop_name	the name used for the corresponding Archimedean copula (generated with cop)

**Author(s)**

Simon-Pierre Gadoury

**Examples**

```
addNode(type = "Child",
        pp = "gamma",
        name_short = "pois",
        name_long = "poisson",
        Laplace = "exp(gamma * (exp(-(z)) - 1))",
        LaplaceInv = "-log(log(z) / gamma + 1)",
        NULL,
        NULL,
        simul = function(n, gamma) rpois(n, gamma),
        cop_name = NULL)

## Construct a bivariate Archimedean copula with the distribution

dist <- POIS(5, 1:2, NULL)
dist@cop(5, 2)
```

```
## Add the distribution in a structure

GEO(0.1, NULL, list(GAMMA(0.5, 1:2, NULL),
                    POIS(5, 3:4, NULL)))
```

AMH

*Construction of an Archimedean Copula Class Object.***Description**

Constructs a AMH Archimedean copula object with a given parameter and dimension.

**Usage**

```
AMH(param, dim = 2L, density = FALSE)
```

**Arguments**

param	parameter of the copula.
dim	dimension of the copula ( $\geq 2$ ), which is, by default, 2.
density	compute the expression of the density of the copulas.

**Details**

Constructs an AMH Archimedean copula object with a given parameter and dimension.

**Value**

An archm S4 class object.

**Author(s)**

Simon-Pierre Gadoury

Clayton

*Construction of an Archimedean Copula Class Object***Description**

Constructs a Clayton Archimedean copula object with a given parameter and dimension.

**Usage**

```
Clayton(param, dim = 2L, density = FALSE)
```

**Arguments**

param	the parameter of the copula.
dim	the dimension of the copula ( $\geq 2$ ), which is, by default, 2.
density	logical. Should the expression of the density of the copula be computed?

**Value**

An archm S4 class object.

**Author(s)**

Simon-Pierre Gadoury

---

Frank

*Construction of an Archimedean Copula Class Object*

---

**Description**

Constructs a Frank Archimedean copula object with a given parameter and dimension.

**Usage**

```
Frank(param, dim = 2L, density = FALSE)
```

**Arguments**

param	parameter of the copula.
dim	dimension of the copula ( $\geq 2$ ), which is, by default, 2.
density	compute the expression of the density of the copulas.

**Details**

Constructs Frank Archimedean copula object with a given parameter and dimension.

**Value**

An archm S4 class object.

**Author(s)**

Simon-Pierre Gadoury

**Description**

The function GAMMA constructs a gamma Child class object for a given parameter and arguments.

**Usage**

```
GAMMA(par, unif, struc = NULL)
```

**Arguments**

par	Dimension of the distribution
unif	Uniform structure, a numeric vector of grouped numbers. i.e. c(1,2,3) is translated as being c(u1, u2, u3).
struc	Nesting structure of the form X(par1, c(i,...), list(Y(par2, c(j,...), NULL), Z(par3, c(k,...), NULL))), where X, Y, and Z are compatible functions (see 'details'). It is to note that if struc is NULL, the function will automatically be of class Child. For continuous distributions (i.e. GAMMA), struc is always NULL.

**Author(s)**

Simon-Pierre Gadoury

**See Also**

Other mother.or.child.class.objects: [GEO](#), [LOG](#)

**Examples**

```
We recall that since GAMMA is continuous, the structure is exclusively a child.
GEO(0.5, NULL, list(GAMMA(1/30, c(5,6), NULL),
                    GEO(0.1, NULL, list(GAMMA(1/30, c(1,2), NULL),
                                       GAMMA(1/30, c(3,4), NULL))))))
```

**Description**

Function to obtain the list of all genetic codes of a structure.

**Usage**

```
GeneticCodes(str)
```

**Arguments**

`str` an object of class `Mother` (the structure)

**Value**

A list of the structure's genetic codes.

**Author(s)**

Simon-Pierre Gadoury

**Examples**

```
## Create the structure
str <- GEO(0.5, NULL, list(GAMMA(1/30, c(5,6), NULL),
                          GEO(0.1, NULL, list(GAMMA(1/30, c(1,2), NULL),
                                              GAMMA(1/30, c(3,4), NULL))))))

## Get the genetic codes
GeneticCodes(str)
```

---

GEO

---

*Construction of a GEO Mother or Child Class Object*


---

**Description**

Constructs either a GEO Mother or Child class object for a given parameter, arguments, and nesting structure.

**Usage**

```
GEO(par, unif, struc)
```

**Arguments**

`par` Dimension of the distribution

`unif` Uniform structure, a numeric vector of grouped numbers.  
i.e. `c(1,2,3)` is translated as being `c(u1, u2, u3)`.

`struc` Nesting structure of the form  
`X(par1, c(i,...), list(Y(par2, c(j,...), NULL), Z(par3, c(k,...), NULL)))`,  
where `X`, `Y`, and `Z` are compatible functions (see 'details'). It is to note that if `struc` is `NULL`, the function will automatically be of class `Child`. For continuous distributions (i.e. `GAMMA`), `struc` is always `NULL`.

**Author(s)**

Simon-Pierre Gadoury

**See Also**

Other `mother.or.child.class.objects`: [GAMMA](#), [LOG](#)

**Examples**

```
GEO(0.5, NULL, list(GAMMA(1/30, c(5,6), NULL),
                    GEO(0.1, NULL, list(GAMMA(1/30, c(1,2), NULL),
                                         GAMMA(1/30, c(3,4), NULL))))))
```

---

Gumbel	<i>Construction of an Archimedean Copula Class Object</i>
--------	---

---

**Description**

Constructs a Gumbel Archimedean copula object with a given parameter and dimension

**Usage**

```
Gumbel(param, dim = 2L)
```

**Arguments**

param	Parameter of the copula
dim	Dimension of the copula ( $\geq 2$ ), which is, by default, 2

**Value**

An archm S4 class object

**Author(s)**

Simon-Pierre Gadoury

---

InvLap	<i>Inverse LST of a Node</i>
--------	------------------------------

---

**Description**

With a specific path and a predefined structure (S4 class of a type 'Mother'), returns the inverse Laplace-Stieltjes Transform expression of the corresponding node with a specific variable.

**Usage**

```
InvLap(code, str, tt = "z", par = "value")
```

**Arguments**

code	the genetic code (numeric vector) of the node (can be a leaf i.e. end by 0)
str	an object of class Mother (the structure)
tt	the output variable to be used ('z' by default)
par	logical. Should the parameters be values ('value') or variables ('variable') ?

Details

For mother nodes, parameters are always called 'gamma' and for child nodes, parameters are always called 'alpha'. Furthermore, to recognize the parameters, the path is inserted at the end. For example, a child node with path (0,2,1) will have the parameter 'alpha021'.

Value

A character string giving the inverse LST of the specified node.

Author(s)

Simon-Pierre Gadoury

See Also

[Lap](#)

Examples

```
str <- GEO(0.1, NULL, list(GAMMA(0.1, 1:2, NULL),
                           GAMMA(0.2, 3:4, NULL)))

InvLap(c(0,2), str, tt = 'z', par = 'value')
```

---

Lap	<i>LST of a Node</i>
-----	----------------------

---

Description

With a specific path and a predefined structure (S4 class of a type 'Mother'), returns the Laplace-Stieltjes Transform expression of the corresponding node with a specific variable.

Usage

```
Lap(code, str, tt = "z", par = "value")
```

Arguments

- code                    Genetic code (numeric vector) of the node (can be a leaf i.e. end by 0)
- str                    Object of class Mother (the structure)
- tt                    Output variable to be used ('z' by default)
- par                    Should the parameters be values ('value') or variables ('variable') ?

Details

For mother nodes, parameters are always called 'gamma' and for child nodes, parameters are always called 'alpha'. Furthermore, to recognize the parameters, the path is inserted at the end. For example, a child node with path (0,2,1) will have the parameter 'alpha021'.



**Value**

A character string giving the LST of the specified node.

**Author(s)**

Simon-Pierre Gadoury

**See Also**

[InvLap](#)

**Examples**

```
str <- GEO(0.1, NULL, list(GAMMA(0.1, 1:2, NULL),
                          GAMMA(0.2, 3:4, NULL)))

Lap(c(0,2), str, tt = 'z', par = 'value')
```

---

LOG

---

*Construction of a LOG Mother or Child Class Object*


---

**Description**

Constructs either a LOG Mother or Child class object for a given parameter, arguments, and nesting structure.

**Usage**

```
LOG(par, unif, struc)
```

**Arguments**

par	Dimension of the distribution
unif	Uniform structure, a numeric vector of grouped numbers. i.e. c(1,2,3) is translated as being c(u1, u2, u3).
struc	Nesting structure of the form X(par1, c(i,...), list(Y(par2, c(j,...), NULL), Z(par3, c(k,...), NULL))), where X, Y, and Z are compatible functions (see 'details'). It is to note that if struc is NULL, the function will automatically be of class Child. For continuous distributions (i.e. GAMMA), struc is always NULL.

**Author(s)**

Simon-Pierre Gadoury

**See Also**

Other mother.or.child.class.objects: [GAMMA](#), [GEO](#)

**Examples**

```
LOG(0.5, NULL, list(GAMMA(1/30, c(5,6), NULL),
                    LOG(0.1, NULL, list(GAMMA(1/30, c(1,2), NULL),
                                         GAMMA(1/30, c(3,4), NULL))))))
```

---

Node	<i>Obtain a node in mother class object</i>
------	---

---

**Description**

Use a path (numeric vector) to obtain a subgroup of a structure (mother class object).

**Usage**

```
Node(path, str)
```

**Arguments**

path	the path of the node (numeric vector).
str	a mother class object (S4).

**Details**

Every node of a mother object (structure) can be identified with a numeric vector that indicates the path used from the root to the node. The vector is the 'path' argument and is used to find specific nodes of a given structure. For a complete explanation, we refer to Cossette et al. (2017).

**Value**

Either a child or mother class object.

**Author(s)**

Simon-Pierre Gadoury

**Examples**

```
# We directly give the path of the desired node.
Node(c(0,2,2), LOG(0.5, NULL, list(GAMMA(1/30, c(5,6), NULL),
                                   LOG(0.1, NULL, list(GAMMA(1/30, c(1,2), NULL),
                                                         GAMMA(1/30, c(3,4), NULL))))))

# Here we provide the path with the GeneticCodes function of this package.
str <- LOG(0.5, NULL, list(GAMMA(1/30, c(5,6), NULL),
                          LOG(0.1, NULL, list(GAMMA(1/30, c(1,2), NULL),
                                                GAMMA(1/30, c(3,4), NULL))))))

Node(GeneticCodes(str)[[3]], str)
```

---

pCompCop	<i>Distribution function of Mother class objects</i>
----------	--

---

**Description**

Distribution function of a Mother class object.

**Usage**

```
pCompCop(str, vector = FALSE, express = TRUE)
```

**Arguments**

str	Object of class Mother
vector	logical. If false, returns a function or a character string with (u_1, u_2, ...) as arguments, else, just (u).
express	logical. If false, returns a function, else, a character string.

**Value**

The distribution function in the form of either a function or a character string.

**Examples**

```
## Create the structure
str <- LOG(0.5, NULL, list(GAMMA(1/30, c(5,6), NULL),
                        LOG(0.1, NULL, list(GAMMA(1/30, c(1,2), NULL),
                        GAMMA(1/30, c(3,4), NULL))))))

## Character string
pCompCop(str, vector = TRUE, express = TRUE)
pCompCop(str, vector = FALSE, express = TRUE)

## Function
pCompCop(str, vector = TRUE, express = FALSE)
pCompCop(str, vector = FALSE, express = FALSE)
```

---

pCop	<i>Distribution function of archm class objects</i>
------	---

---

**Description**

Distribution function of an Archimedean copula (archm) class object.

**Usage**

```
pCop(copula, vector = FALSE, express = TRUE)
```

**Arguments**

copula	an Archimedean copula (archm) class object.
vector	logical. If false, returns a function or a character string with (u_1, u_2, ..., u_dim) as arguments, else, just (u).
express	logical. If false, returns a function, else, a character string.

**Value**

The distribution function in the form of either a function or a character string.

**Author(s)**

Simon-Pierre Gadoury

**See Also**

[rCop](#), [Clayton](#), [AMH](#), [Gumbel](#), [Frank](#)

**Examples**

```
cop <- Clayton(5, 2)
pCop(cop, vector = TRUE, express = TRUE)
pCop(cop, vector = FALSE, express = TRUE)
```

---

rCompCop

*Density, Cdf, and Random Number Generator for Copulas Constructed Through Compounding*

---

**Description**

Random number generator for copulas constructed through compounding.

**Usage**

```
rCompCop(n, str)
```

**Arguments**

n	the number of realisations
str	an object of class Mother

**Value**

A numeric matrix of sampled data from the structure

**Author(s)**

Simon-Pierre Gadoury

**Examples**

```
## Create the structure
str <- GEO(0.1, 1, list(GAMMA(0.2, 2:3, NULL),
                       GEO(0.3, 4:5, NULL)))

## Sample from the structure
rCompComp(1000, str)
```

rCop

*Random number generator for Archimedean copula class objects***Description**

Random number generator for archm class objects.

**Usage**

```
rCop(n, copula)
```

**Arguments**

n	Number of realisations
copula	An Archimedean copula (archm) class object

**Details**

For bivariate archm copula objects, the function uses the conditional approach. As for dimensions higher than 2, the Marshall-Olkin (1988) approach is chosen instead.

**Value**

A numeric matrix containing the samples.

**Author(s)**

Simon-Pierre Gadoury

**See Also**

[pCop](#), [Clayton](#), [AMH](#), [Frank](#), [Gumbel](#)

**Examples**

```
## Create the trivariate archm copula object
cop <- Clayton(5, 3)

## Generate the samples
res <- rCop(10000, cop)

## Plot the values
pairs(res, pch = 16, cex = 0.7)
```



# Index

addNode, [2](#)  
AMH, [3](#), [12](#), [13](#)  
  
Clayton, [3](#), [12](#), [13](#)  
  
Frank, [4](#), [12](#), [13](#)  
  
GAMMA, [5](#), [6](#), [9](#)  
GeneticCodes, [5](#)  
GEO, [5](#), [6](#), [9](#)  
Gumbel, [7](#), [12](#), [13](#)  
  
InvLap, [7](#), [9](#)  
  
Lap, [8](#), [8](#)  
LOG, [5](#), [6](#), [9](#)  
  
Node, [10](#)  
  
pCompCop, [11](#)  
pCop, [11](#), [13](#)  
  
rCompCop, [12](#)  
rCop, [12](#), [13](#)  
rStruc, [14](#)