Package 'Compounding'

February 19, 2015

Type Package

Title Computing Continuous Distributions

Version 1.0.2

Date 2012-10-19

Author Bozidar V. Popovic, Saralees Nadarajah, Miroslav M. Ristic

Maintainer Bozidar V. Popovic <bozidarpopovic@gmail.com>

Depends R (>= 2.12.0), hypergeo

Description Computing Continuous Distributions Obtained by Compounding a Continuous and a Discrete Distribution

License GPL (>= 2)

Repository CRAN

Date/Publication 2013-02-09 18:31:03

NeedsCompilation no

R topics documented:

`Compounding-package` *Calculation of the main characteristics of compounding distribution.*

Description

Package Compounding provides values of the pdf, cdf and hazard rate functions of the compounding distribution. Also it is possible to draw random sample and to compute main characteristics of the compounding distribution.

Compound distributions can be characterized as follows:

Suppose a device has an unknown number, N, of initial defects of same kind (for example, a number of semiconductors from a defective lot). Suppose X_i's represent their lifetimes and that each defect can be detected only after causing failure. Then the time to the first failure of the device is X = min(X_1, X_2, ldots, X_N).

Suppose a parallel system has N components. Let X_1, X_2, ldots, X_N denote their lifetimes. The system will fail as soon as any one of the components fails. The system's lifetime is X = min (X_1, X_2, ldots, X_N).

Details

Package:	Compounding
Type:	Package
Version:	1.0.1
Date:	2012-10-19

License:	What license is it under?
LazyLoad:	yes

In this package we give some programs for working with continuous distributions obtained by compounding continuous distributions with discrete distributions. The programs compute values of cumulative distribution function, probability density function, quantile function and hazard rate function, generate random samples from a population with compounding distribution, and compute mean, variance, skewness and kurtosis of a random variable with a compounding distribution. We consider 24 discrete distributions which can be compounded with any continuous distribution implemented in R.

Author(s)

S.Nadarajah, B. V. Popovic, M. M. Ristic,

Maintainer: B.V. Popovic Email: bozidarpopovic@gmail.com

References

S. Nadarajah, B.V. Popovic, M.M. Ristic (2012) Compounding: an R package for computing continuous distributions obtained by compounding a continuous and a discrete distribution, Computational Statistics, DOI 10.1007/s00180-012-0336-y, http://www.springerlink.com/content/6r464013w6mp3545/

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
parentD<-"exp"
compoundD<-"poisson"
params<-2.5
x<-0.5
k<-2
dCompound(x,parentD,compoundD,compoundDist,params)
qCompound(x,parentD,compoundD,compoundDist,params)
hCompound(x,parentD,compoundD,compoundDist,params)
momentCompound(k, parentD, compoundD, compoundDist,params)
```

compoundDist *Compound distribution*

Description

This list is necessary to use functions pCompound, dCompound, qCompound, rCompound, hCompound, momentCompound, meanCompound, varCompound, skewCompound, kurtCompound. This list defines the range of discrete distributions to which the package was defined.

Usage

```
data(compoundDist)
```

Format

The format is: chr [1:24] "geometric" "poisson" "binomial" "negativebinomial" "lorarithmic" "binomialbinomial" "binomialpoisson" "poissonbinomial" "neymantypea" "neymantypeb" "neymantypec" "polyaaepli" "poissonpascal" "pascalpoisson" "logarithmicbinomial" "logarithmicpoisson" "poissonlindley" "hyperpoisson" "yule" "waring" "kattitypeh1" "kattitypeh2" "hypergeometric" "thomas"

Examples

```
## This list should be definned as follows
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
```

dCompound *function dCompound*

Description

Function dCompound calculates value of the pdf of the random variable X.

Usage

```
dCompound(x,parent,compound,compoundDist,params,...)
```

Arguments

x	vector of quantiles
parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B.V. Popovic, M.M. Ristic

References

S. Nadarajah, B.V. Popovic, M.M. Ristic (2012) Compounding: an R package for computing continuous distributions obtained by compounding a continuous and a discrete distribution, Computational Statistics, DOI 10.1007/s00180-012-0336-y, http://www.springerlink.com/content/6r464013w6mp3545/

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
q<-0.5
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
dCompound(q,parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)


## The function is currently defined as

dCompound <- function(x,parent,compound,compoundDist,params,...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
```

```
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
    xval <- real(length(x))
    f <- get(paste("d", parent, sep = ""), mode = "function")
    F <- get(paste("p", parent, sep = ""), mode = "function")
    phi <- get(paste("pgf",compound,sep=""), mode = "function")
    phiD <- get(paste("pgfD",compound,sep=""), mode = "function")
    xval <- phiD(1-F(x,...),params)*f(x,...)/(1-phi(0,params))
    return(xval)
}
```

---

hCompound *function hCompound*

Description

Function hCompound calculates value of the hazard rate function of the random variable X.

Usage

```
hCompound(x,parent,compound,compoundDist,params,...)
```

Arguments

x	vector of quantiles
parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

S. Nadarajah, B.V. Popovic, M.M. Ristic (2012) Compounding: an R package for computing continuous distributions obtained by compounding a continuous and a discrete distribution, Computational Statistics, DOI 10.1007/s00180-012-0336-y, http://www.springerlink.com/content/6r464013w6mp3545/

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
q<-0.5
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
hCompound(q,parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as
hCompound <- function(x, parent, compound,params, ...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
    dCompound(x,parent,compound,compoundDist,params,...)/(1-pCompound(x,parent,compound,
    compoundDist,params,...))
}
```

---

kurtCompound *function kurtCompound*

Description

Function kurtCompound calculates kurtosis of the random variable X.

Usage

```
kurtCompound(parent,compound,compoundDist,params,...)
```

Arguments

parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

S. Nadarajah, B.V. Popovic, M.M. Ristic (2012) Compounding: an R package for computing continuous distributions obtained by compounding a continuous and a discrete distribution, Computational Statistics, DOI 10.1007/s00180-012-0336-y, http://www.springerlink.com/content/6r464013w6mp3545/

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
kurtCompound(parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as
kurtCompound <- function(parent,compound,compoundDist,params,...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
```

```
   m1 <- meanCompound(parent,compound,compoundDist,params,...)
   m3 <- momentCompound(3,parent,compound,compoundDist,params,...)
   m4 <- momentCompound(4,parent,compound,compoundDist,params,...)
   sig2 <- varCompound(parent,compound,compoundDist,params,...)
   return((m4-4*m1*m3+6*m1^2*sig2+3*m1^4)/sig2^2)
}
```

---

meanCompound *function meanCompound*

Description

Function meanCompound calculates mean value of the random variable X.

Usage

```
meanCompound(parent,compound,compoundDist,params,...)
```

Arguments

parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

S. Nadarajah, B.V. Popovic, M.M. Ristic (2012) Compounding: an R package for computing continuous distributions obtained by compounding a continuous and a discrete distribution, Computational Statistics, DOI 10.1007/s00180-012-0336-y, http://www.springerlink.com/content/6r464013w6mp3545/

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
meanCompound(parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as

meanCompound <- function(parent,compound,compoundDist,params,...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
    return(momentCompound(1,parent,compound,compoundDist,params,...))
}
```

`momentCompound` *function momentCompound*

Description

Function momentCompound calculates moments of the random variable X.

Usage

```
momentCompound(k,parent,compound,compoundDist,params,...)
```

Arguments

`k`	integer number representing moment's order
`parent`	name of the parent distribution. It can be any continuous distribution supported by R.
`compound`	name of the compound distribution. It can be any discrete distribution supported by this package.
`compoundDist`	list of available compounding distributions
`params`	Parameter or list of parameters of compounding distribution.

... Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

S. Nadarajah, B.V. Popovic, M.M. Ristic (2012) Compounding: an R package for computing continuous distributions obtained by compounding a continuous and a discrete distribution, Computational Statistics, DOI 10.1007/s00180-012-0336-y, http://www.springerlink.com/content/6r464013w6mp3545/

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
k<-3
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
momentCompound(k,parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as
function(k, parent, compound, compoundDist,params, ...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
    Finv <- get(paste("q", parent, sep = ""), mode = "function")
    phi <- get(paste("pgf",compound,sep=""), mode = "function")
    phiD <- get(paste("pgfD",compound,sep=""), mode = "function")
    fint <- function(x) phiD(1-x,params)*(Finv(x,...))^k/(1-phi(0,params))
    return(integrate(fint,lower=0,upper=1)$value)
}
```

pCompound *function pCompound*

Description

Function pCompound calculates values of the cdf of the random variable X.

Usage

```
pCompound(q,parent,compound,compoundDist,params,...)
```

Arguments

q	vector of quantiles
parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

S. Nadarajah, B.V. Popovic, M.M. Ristic (2012) Compounding: an R package for computing continuous distributions obtained by compounding a continuous and a discrete distribution, Computational Statistics, DOI 10.1007/s00180-012-0336-y, http://www.springerlink.com/content/6r464013w6mp3545/

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
```

```
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
q<-0.5
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
pCompound(q,parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as

pCompound <- function(q,parent,compound,compoundDist,params,...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
    xval <- real(length(q))
    F <- get(paste("p",parent,sep=""), mode = "function")
    phi <- get(paste("pgf",compound,sep=""), mode = "function")
    xval <- (1-phi(1-F(q,...),params))/(1-phi(0,params))
    return(xval)
}
```

---

pgfbinomial *Function pgfbinomial*

Description

This function calculates value of the pgf of the binomial distribution.

Usage

```
pgfbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. Otherwise pgf diverges.
params	List of the parameters of the binomial distribution, such that params<-c(n,theta), where n is the size, while theta is the probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(3,.2)
pgfbinomial(.5,params)


##The function is currently defined as

pgfbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
       n<-params[1]
       theta<-params[2]
if ((theta>=1)|(theta<=0))
    stop ("Parameter theta belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    (1-theta+theta*s)^n
}
```

`pgfbinomialbinomial` *Function pgfbinomialbinomial*

Description

This function calculates value of the pgf of the binomial-binomial distribution.

Usage

```
pgfbinomialbinomial(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the binomial-binomial distribution, such that params<-c(p1,p2,m,n), where theta is the positive number, p1, p2 are the probabilities, and m, n are the positive integers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.3,.2,4,5)
pgfbinomialbinomial(.5,params)


## The function is currently defined as

pgfbinomialbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<4) stop("At least one value in params is missing")
if (length(params)>4) stop("The length of params is 4")
    p1<-params[1]
    p2<-params[2]
    m<-params[3]
    n<-params[4]
if ((p1>=1)|(p1<=0))
stop ("Parameter p1 belongs to the interval (0,1)")
if ((p2>=1)|(p2<=0))
stop ("Parameter p2 belongs to the interval (0,1)")
if (m<0)
stop("Parameter m must be positive integer")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    (1-p1+p1*(1-p2+p2*s)^n)^m
}
```

---

pgfbinomialpoisson *Function pgfbinomialpoisson*

Description

This function calculates value of the pgf of the binomial-Poisson distribution.

Usage

```
pgfbinomialpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the binomial-Poisson distribution, such that params<-c(theta,p,n), where theta is the positive number, p is the probability, and n is the positive integer.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,.5,2)
pgfbinomialpoisson(.5,params)


## The function is currently defined as
pgfbinomialpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3) stop("At least one value in params is missing")
if (length(params)>3) stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if (theta<=0)
stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
   (1-p+p*exp(theta*(s-1)))^n
}
```

pgfDbinomial *Function pgfDbinomial*

Description

This function calculates value of the pgf's first derivative of the binomial distribution.

Usage

```
pgfDbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the binomial distribution, such that params<-c(n,theta), where n is size and theta is probability.

Value

1 0.59895

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,.9)
pgfDbinomial(.5,params)


## The function is currently defined as

pgfDbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
     stop("At least one value in params is missing")
if (length(params)>2)
     stop("The length of params is 2")
```

```
       n<-params[1]
       theta<-params[2]
if ((theta>=1)|(theta<=0))
     stop ("Parameter theta belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    n*theta*(1-theta+theta*s)^(n-1)
}
```

---

pgfDbinomialbinomial *Function pgfDbinomialbinomial*

Description

This function calculates value of the pgf's firts derivative of the binomial-binomial distribution.

Usage

```
pgfDbinomialbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Poisson-binomial distribution, such that params<-c(p1,p2,m,n), where theta is the positive number, p1, p2 are the probabilities, and m,n are the positive integers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,.9,5,7)
pgfDbinomialbinomial(.5,params)


## The function is currently defined as
```

```
pgfDbinomialbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<4)
    stop("At least one value in params is missing")
if (length(params)>4)
    stop("The length of params is 4")
    p1<-params[1]
    p2<-params[2]
    m<-params[3]
    n<-params[4]
if ((p1>=1)|(p1<=0))
    stop ("Parameter p1 belongs to the interval (0,1)")
if ((p2>=1)|(p2<=0))
    stop ("Parameter p2 belongs to the interval (0,1)")
if (m<0)
    stop("Parameter m must be positive integer")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    m*n*p1*p2*(1-p2+p2*s)^(n-1)*(1-p1+p1*(1-p2+p2*s)^n)^(m-1)
}
```

---

pgfDbinomialpoisson *Function pgfDbinomialpoisson*

Description

This function calculates value of the pgf's first derivative of the binomial-Poisson distribution.

Usage

```
pgfDbinomialpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the binomial-Poisson distribution, such that params<-c(theta,p,n), where theta is the positive number, p is the probability, and n is the positive integer.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,.9,5)
pgfDbinomialpoisson(.5,params)


## The function is currently defined as

pgfDbinomialpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
    stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    n*theta*p*exp(theta*(s-1))*(1-p+p*exp(theta*(s-1)))^(n-1)
}
```

`pgfDgeometric` *Function pgfDgeometric*

Description

This function calculates value of the pgf's first derivative of the geometric distribution.

Usage

```
pgfDgeometric(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Parameter of the geometric distribution, such that params<-theta, where theta is the probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-.1
pgfDgeometric(.5,params)

## The function is currently defined as

pgfDgeometric <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
   warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
    stop("The length of params is 1")
    theta<-params[1]
if ((theta>=1)|(theta<=0))
    stop ("Parameter of geometric distribution must belong  to the interval (0,1)")
    theta*(1-theta)/(1-(1-theta)*s)^2
}
```

---

pgfDhypergeometric *Function pgfDhypergeometric*

Description

This function calculates value of the pgf's first derivative of the hypergeometric distribution.

Usage

```
pgfDhypergeometric(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the hypergeometric distribution, such that params<-c(m,n,p), where m is the number of white balls in the urn, n is the number of black balls in the urn, must be less or equal than m, and p is the probability.

Value

1 0.6

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,3,.2)
pgfDhypergeometric(.5,params)

## The function is currently defined as

pgfDhypergeometric <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
     stop("At least one value in params is missing")
if (length(params)>3)
     stop("The length of params is 3")
    m<-params[1]
    n<-params[2]
    p<-params[3]

if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
if (m<0)
     stop("Parameter m must be positive")
 if(!(abs(m-round(m))<.Machine$double.eps^0.5))
```

```
stop("Parameter m must be positive integer")
if ((p>=1)|(p<=0))
     stop ("Parameter p belongs to the interval (0,1)")
if (m<n)
     stop ("Parameter m is greater or equal then n ")
   n*p*Re(hypergeo(1-n,1-m*p,1-m,1-s))
}
```

---

pgfDhyperpoisson *Function pgfDhyperpoisson.*

Description

This function calculates value of the pgf's first derivative of the hyper Poisson distribution.

Usage

```
pgfDhyperpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the hyper Poisson distribution, such that params<-c(theta,lambda), where both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.7,3)
pgfDhyperpoisson(.5,params)

## The function is currently defined as

pgfDhyperpoisson <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
```

```
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")
  theta*genhypergeo(2,lambda+1,theta*s)/(lambda*genhypergeo(1,lambda,theta))
}
```

---

pgfDkattitypeh1 *Function pgfDkattitypeh1*

Description

This function calculates value of the pgf's first derivative of the Katti type H1 distribution.

Usage

```
pgfDkattitypeh1(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1\]. In the opposite pgf diverges.
params	List of the parameters of the Katti type H1 distribution, such that params<-c(theta,a,b), where all parameter are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,9,5)
pgfDkattitypeh1(.5,params)

## The function is currently defined as

pgfDkattitypeh1 <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
    stop("The length of params is 3")
    theta<-params[1]
    a<-params[2]
    b<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (a<=0)
    stop ("Parameter a must be positive")
if (b<=0)
    stop ("Parameter b must be positive")
a*theta/(a+b)*genhypergeo(a+1,a+b+1,theta*(s-1))
}
```

---

pgfDkattitypeh2 *Function pgfDkattitypeh2*

Description

This function calculates value of the pgf's derivative of the Katti type H2 distribution.

Usage

```
pgfDkattitypeh2(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Katti type H2 distribution, such that params<-c(theta,a,b,k), where all parameter are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,9,5,7)
pgfDkattitypeh2(.5,params)


## The function is currently defined as

pgfDkattitypeh2 <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<4)
   stop("At least one value in params is missing")
if (length(params)>4)
   stop("The length of params is 4")
    theta<-params[1]
    a<-params[2]
    b<-params[3]
    k<-params[4]
if (theta<=0)
  stop ("Parameter theta must be positive")
if (a<=0)
  stop ("Parameter a must be positive")
if (b<=0)
  stop ("Parameter b must be positive")
if (k<=0)
  stop ("Parameter k must be positive")
a*k*theta/(a+b)*Re(hypergeo(k+1,a+1,a+b+1,theta*(s-1)))
}
```

`pgfDlogarithmic` *Function pgfDlogarithmic.*

Description

This function calculates value of the pgf's first derivative of the logarithmic distribution.

Usage

```
pgfDlogarithmic(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1\]. In the opposite pgf diverges.
params	Represents paramemeter of logarithmic distribution, such that params<-theta, where theta is the probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-.7
pgfDlogarithmic(.5,params)

## The function is currently defined as

pgfDlogarithmic <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
     stop("The length of params is 1")
     theta<-params[1]
if ((theta>=1)|(theta<=0))
     stop ("Parameter theta belongs to the interval (0,1)")
    -(1-theta)/((1-(1-theta)*s)*log(theta))
}
```

---

pgfDlogarithmicbinomial

Function pgfDlogarithmicbinomial.

Description

This function calculates value of the pgf's first derivative of the logarithmic-binomial distribution.

Usage

```
pgfDlogarithmicbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the logarithmic-binomial distribution, such that params<-c(p1,p2,m,n), where p1, p2 are the probabilities, m, n are the positive integers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.9,.7,4)
pgfDlogarithmicbinomial(.5,params)

## The function is currently defined as

pgfDlogarithmicbinomial <- function(s,params) {
    k<-s[abs(s)>1]
if (length(k)>0)
   warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
   stop("At least one value in params is missing")
if (length(params)>3)
   stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if ((theta>=1)|(theta<=0))
    stop ("Parameter theta belongs to the interval (0,1)")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
-n*p*(1-theta)/log(theta)*(1-p+p*s)^(n-1)*(1-(1-theta)*(1-p+p*s)^n)^(-1)
}
```

pgfDlogarithmicpoisson

Function pgfDlogarithmicpoisson

Description

This function calculates value of the pgf's first derivative of the logarithmic-Poisson distribution.

Usage

```
pgfDlogarithmicpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the logarithmic-Poisson distribution, such that params<-c(theta,lambda), where theta is the probability, lambda is the positive number.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.9,7)
pgfDlogarithmicpoisson(.5,params)

## The function is currently defined as

pgfDlogarithmicpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
     theta<-params[1]
    lambda<-params[2]
if ((theta>=1)|(theta<=0))
     stop ("Parameter theta belongs to the interval (0,1)")
```

```
if (lambda<=0)
     stop ("Parameter lambda must be positive")
   -lambda*(1-theta)/log(theta)*exp(lambda*(s-1))/(1-(1-theta)*exp(lambda*(s-1)))
}
```

---

pgfDnegativebinomial *Function pgfDnegativebinomial*

Description

This function calculates value of the pgf of the negative binomial distribution.

Usage

```
pgfDnegativebinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the negative binomial distribution, such that params<-c(theta,k), where theta is the probability, k is the positive number.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.3,.2)
pgfDnegativebinomial(.5,params)

## The function is currently defined as

pgfDnegativebinomial <- function(s,params) {
  k <- s[abs(s)>1]
  if (length(k)>0) warning("Some  elements of the vector s are out of interval [-1,1]")
  if (length(params)<2) stop("At least one value in params is missing")
  if (length(params)>2) stop("The length of params is 2")
  theta <- params[1]
  k <- params[2]
  if ((theta>=1) | (theta<=0)) stop ("Parameter theta belongs to the interval (0,1)")
  if (k<=0) stop("Parameter k must be positive")
```

```
  k*(1-theta)*theta^k/(1-(1-theta)*s)^(k+1)
}
```

---

pgfDneymantypea *Function pgfDneymantypea*

Description

This function calculates value of the pgf's first derivative of the Neyman type A distribution.

Usage

```
pgfDneymantypea(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type A distribution, such that params<-c(theta,lambda), where both parameters are positive numbers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,5)
pgfDneymantypea(.5,params)


## The function is currently defined as

pgfDneymantypea <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
```

```
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")
    lambda*theta*exp(theta*(s-1))*exp(lambda*(exp(theta*(s-1))-1))
}
```

---

pgfDneymantypeb *Function pgfDneymantypeb*

Description

This function calculates value of the pgf's first derivative of the Neyman type B distribution.

Usage

```
pgfDneymantypeb(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type B distribution, such that params<-c(theta,lambda), where both parameters are positive numbers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,5)
pgfDneymantypeb(.5,params)


## The function is currently defined as

pgfDneymantypeb <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
```

```
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")
    theta*lambda/2*genhypergeo(2,3,theta*(s-1))*pgfneymantypeb(s,params)
}
```

---

pgfDneymantypec *Function pgfDneymantypec*

Description

This function calculates value of the pgf's first derivative of the Neyman type C distribution.

Usage

```
pgfDneymantypec(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type C distribution, such that params<-c(theta,lambda), where both parameters are positive numbers

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,5)
pgfDneymantypec(.5,params)


## The function is currently defined as

pgfDneymantypec <- function(s,params) {
    require(hypergeo)
     k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")
    theta*lambda/3*genhypergeo(2,4,theta*(s-1))*pgfneymantypec(s,params)
}
```

`pgfDpascalpoisson` *Function pgfDpascalpoisson*

Description

This function calculates value of the pgf's first derivative of the Pascal Poisson distribution.

Usage

```
pgfDpascalpoisson(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be form interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Pascal Poisson distribution, such that params<-c(theta,mu, a), where all parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(3,2,.5)
pgfDpascalpoisson(.5,params)


## The function is currently defined as

pgfDpascalpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
     stop("At least one value in params is missing")
if (length(params)>3)
     stop("The length of params is 3")
    theta<-params[1]
    mu<-params[2]
    a<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (mu<=0)
    stop ("Parameter mu must be positive")
if (a<=0)
    stop ("Parameter a must be positive")
    mu*exp(theta*(s-1))*(1+mu/(a*theta)-mu/(a*theta)*exp(theta*(s-1)))^(-a-1)
}
```

---

pgfDpoisson *Function pgfDpoisson.*

Description

This function calculates value of the pgf's first derivative of the Poisson distribution.

Usage

```
pgfDpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be form interval [-1,1]. In the opposite pgf diverges.
params	Positive parameter of the Poisson distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-7
pgfDpoisson(.5,params)

## The function is currently defined as

pgfDpoisson <- function(s,params) {
   k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1) stop("The length of params is 1")
    theta<-params[1]
if (theta<=0)
    stop ("Parameter of Poisson distribution must be positive")
    theta*exp(theta*(s-1))
}
```

`pgfDpoissonbinomial` *Function pgfDpoissonbinomial.*

Description

This function calculates value of the pgf of the Poisson-binomial distribution.

Usage

```
pgfDpoissonbinomial(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Poisson-binomial distribution, such that params<-c(theta,p,n), where theta is the positive number, p is the probability, n is the positive integer.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(2.6,.7,3)
pgfDpoissonbinomial(.5,params)


## The function is currently defined as

pgfDpoissonbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
   warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
   stop("At least one value in params is missing")
if (length(params)>3)
   stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    n*theta*p*(1-p+p*s)^(n-1)*exp(theta*((1-p+p*s)^n-1))
}
```

---

pgfDpoissonlindley *Function pgfDpoissonlindley.*

Description

This function calculates value of the pgf's first derivative of the Poisson-Lindley distribution.

Usage

```
pgfDpoissonlindley(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf divegrates.
params	Positive parameter of the Poisson-Lindley distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-5
pgfDpoissonlindley(.5,params)

## The function is currently defined as

pgfDpoissonlindley <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
    stop("The length of params is 1")
    theta<-params[1]
if (theta<=0)
    stop ("Parameter lambda must be positive")
    (2/(theta+1-s)-1/(theta+2-s))*pgfpoissonlindley(s,params)
}
```

---

pgfDpoissonpascal *Function pgfDpoissonpascal.*

Description

This function calculates value of the pgf's first derivative of the Poisson-Pascal distribution.

Usage

```
pgfDpoissonpascal(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Poisson-Pascal distribution, such that params<-c(theta,p,k), where all parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.4,.3)
pgfDpoissonpascal(.5,params)

## The function is currently defined as

pgfDpoissonpascal <- function(s,params) {
m<-s[abs(s)>1]
if (length(m)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
    stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    k<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (p<=0)
    stop ("Parameter lambda must be positive")
if (k<=0)
    stop ("Parameter k must be positive")
    theta*k*p*(1+p-p*s)^(-k-1)*exp(theta*((1+p-p*s)^(-k)-1))
}
```

---

pgfDpolyaaeppli *Function pgfDpolyaaeppli*

Description

This function calculates value of the pgf's first derivative of the Polya Aeppli distribution.

Usage

```
pgfDpolyaaeppli(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Polya Aeppli distribution, such that params<-c(theta,p), where theta is the positive number, and p is the probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.4)
pgfDpolyaaeppli(.5,params)


## The function is currently defined as

pgfDpolyaaeppli <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    p<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")
    theta*(1-p)/(1-p*s)^2*exp(theta/p*((1-p)/(1-p*s)-1))
}
```

pgfDthomas *Function pgfDthomas*

Description

This function calculates value of the pgf of the Thomas distribution.

Usage

```
pgfDthomas(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf divegrates.
params	List of the parameters of the Thomas distribution, such that params<-c(lambda,theta), where both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.4)
pgfDthomas(.5,params)

## The function is currently defined as

pgfDthomas <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    lambda<-params[1]
    theta<-params[2]
if (lambda<=0)
    stop ("Parameter lambda must be positive")
if (theta<=0)
```

```
    stop ("Parameter theta must be positive")
    lambda*(1+theta*s)*exp(theta*(s-1))*exp(lambda*(s*exp(theta*(s-1))-1))
}
```

---

pgfDwaring *Function pgfDwaring*

Description

This function calculates value of the pgf's first derivative of the Waring distribution.

Usage

```
pgfDwaring(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Waring distribution, such that params<-c(c,a), where $c>a$ and both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.8,.4)
pgfDwaring(.5,params)

## The function is currently defined as

pgfDwaring <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
```

```
if (length(params)>2)
    stop("The length of params is 2")
    cc<-params[1]
    a<-params[2]
if (cc<=0)
    stop ("Parameter c must be positive")
if (a<=0)
    stop ("Parameter a must be positive")

    a*(cc-a)/(cc*(cc+1))*Re(hypergeo(2,a+1,cc+2,s))
}
```

---

pgfDyule *Function pgfDyule*

Description

This function calculates value of the pgf's first derivative of the Yule distribution.

Usage

```
pgfDyule(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf divegrates.
params	Postive parameter of the Yule distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-.3
pgfDyule(.5,params)

## The function is currently defined as
pgfDyule <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
    stop("The length of params is 1")
    theta<-params[1]
if (theta<=0)
    stop ("Parameter theta must be positive")

    theta/((theta+1)*(theta+2))*Re(hypergeo(2,2,theta+3,s))
}
```

---

pgfgeometric *Function pgfgeometric*

Description

This function calculates value of the pgf of the geometric distribution.

Usage

```
pgfgeometric(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Parameter of the geometric distribution, such that params<-theta, where theta is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-0.4
pgfgeometric(.2,params)

## The function is currently defined as

pgfgeometric <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1) stop("The length of params is 1")
 theta<-params[1]
if ((theta>=1)|(theta<=0))
stop ("Parameter of geometric distribution must belong  to the interval (0,1)")
    theta/(1-(1-theta)*s)
}
```

---

pgfhypergeometric *Function pgfhypergeometric*

Description

This function calculates value of the pgf of the hypergeometric distribution.

Usage

```
pgfhypergeometric(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the hypergeometric distribution, such that params<-c(m,n,p) where m is the number of white balls in the urn, n is the number of black balls in the urn, must be less or equal than m, and p is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,3,.5)
pgfhypergeometric(.5,params)


## The function is currently defined as

pgfhypergeometric <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
    stop("The length of params is 3")
    m<-params[1]
    n<-params[2]
    p<-params[3]
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
if (m<0)
     stop("Parameter m must be positive")
 if(!(abs(m-round(m))<.Machine$double.eps^0.5))
stop("Parameter m must be positive integer")
if ((p>=1)|(p<=0))
   stop ("Parameter p belongs to the interval (0,1)")
if (m<n)
   stop ("Parameter m is greater or equal then n ")
    Re(hypergeo(-n,-m*p,-m,1-s))
}
```

---

pgfhyperpoisson *Function pgfhyperpoisson*

Description

This function calculates value of the pgf of the hyper Poisson distribution.

Usage

```
pgfhyperpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the hyperpoisson distribution, such that params<-c(theta,lambda), where both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,.3)
pgfhyperpoisson(.5,params)


## The function is currently defined as

pgfhyperpoisson <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")

    theta<-params[1]
    lambda<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")
genhypergeo(1,lambda,theta*s)/genhypergeo(1,lambda,theta)
}
```

pgfIbinomial *Function pgfIbinomial*

Description

This function calculates value of the pgf's inverse of the binomial distribution.

Usage

```
pgfIbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the binomial distribution, such that params<-c(n,theta), where n is the size, and theta is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,.9)
pgfIbinomial(.5,params)

## The function is currently defined as

pgfIbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
   stop("At least one value in params is missing")
if (length(params)>2)
   stop("The length of params is 2")
   n<-params[1]
   theta<-params[2]
if ((theta>=1)|(theta<=0))
   stop ("Parameter theta belongs to the interval (0,1)")
if (n<0)
```

```
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    (s^(1/n)-1+theta)/theta
}
```

---

pgfIbinomialbinomial *Function pgfIbinomialbinomial*

Description

This function calculates value of the pgf's inverse of the binomial-binomial distribution.

Usage

```
pgfIbinomialbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Poisson-binomial distribution, such that params<-c(p1,p2,m,n), where theta is positive number, p1, p2 are probabilities, and m,n are positive integers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,.9,5,7)
pgfIbinomialbinomial(.5,params)


## The function is currently defined as

pgfIbinomialbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<4)
```

```
    stop("At least one value in params is missing")
if (length(params)>4)
    stop("The length of params is 4")
    p1<-params[1]
    p2<-params[2]
    m<-params[3]
    n<-params[4]
if ((p1>=1)|(p1<=0))
    stop ("Parameter p1 belongs to the interval (0,1)")
if ((p2>=1)|(p2<=0))
    stop ("Parameter p2 belongs to the interval (0,1)")
if (m<0)
    stop("Parameter m must be positive integer")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    zval<-(s^(1/m)-1+p1)/p1
    (zval^(1/n)-1+p2)/p2
}
```

---

pgfIbinomialpoisson *Function pgfIbinomialpoisson*

Description

This function calculates value the pgf's inverse of the binomial-Poisson distribution.

Usage

```
pgfIbinomialpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the binomial-Poisson distribution, such that params<-c(theta,p,n), where theta is positive number, p is probability, and n is positive integer.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,.9,5)
pgfIbinomialpoisson(.5,params)

## The function is currently defined as

pgfIbinomialpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
   warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
   stop("At least one value in params is missing")
if (length(params)>3)
   stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if (theta<=0)
   stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
   stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    zval<-(s^(1/n)-1+p)/p
    1+log(zval)/theta
}
```

---

pgfIgeometric *Function pgfIgeometric*

Description

This function calculates value of the pgf's inverse of the geometric distribution.

Usage

```
pgfIgeometric(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Parameter of the geometric distribution, such that params<-theta, where theta is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-.1
pgfIgeometric(.5,params)

## The function is currently defined as

pgfIgeometric <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
     stop("The length of params is 1")
   theta<-params[1]
if ((theta>=1)|(theta<=0))
     stop ("Parameter of geometric distribution must belong  to the interval (0,1)")
    (s-theta)/((1-theta)*s)
}
```

---

pgfIhypergeometric *Function pgfIhypergeometric*

Description

This function calculates value of the pgf's inverse of the hypergeometric distribution.

Usage

```
pgfIhypergeometric(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the hypergeometric distribution, such that params<-c(m,n,p), where m is the number of white balls in the urn, n is the number of black balls in the urn, must be less or equal than m, and p is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,3,.2)
pgfIhypergeometric(.5,params)

## The function is currently defined as

pgfIhypergeometric <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfhypergeometric(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

`pgfIhyperpoisson` *Function pgfIhyperpoisson*

Description

This function calculates value of the pgf's inverse of the hyper Poisson distribution.

Usage

```
pgfIhyperpoisson(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the hyperpoisson distribution, such that params<-c(theta,lambda), where both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.7,3)
pgfIhyperpoisson(.9,params)

##The function is currently defined as

pgfIhyperpoisson <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfhyperpoisson(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

---

pgfIkattitypeh1 *Function pgfIkattitypeh1*

Description

This function calculates value of the pgf's inverse of the Katti type H1 distribution.

Usage

```
pgfIkattitypeh1(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Katti type H1 distribution, such that params<-c(theta,a,b), where all parameter are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,9,5)
pgfIkattitypeh1(.9,params)


## The function is currently defined as

pgfIkattitypeh1 <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfkattitypeh1(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

---

pgfIkattitypeh2 *Function pgfIkattitypeh2*

Description

This function calculates value of the pgf's inverse of the Katti type H2 distribution.

Usage

```
pgfIkattitypeh2(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Katti type H2 distribution, such that params<-c(theta,a,b,k), where all parameter are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,9,5,7)
pgfIkattitypeh2(.9,params)


 ## The function is currently defined as
pgfIkattitypeh2 <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfkattitypeh2(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

`pgfIlogarithmic` *Function pgfIlogarithmic*

Description

This function calculates value of the pgf's inverse of the logarithmic distribution.

Usage

```
pgfIlogarithmic(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	Paramemeter of logarithmic distribution, such that params<-theta, where theta is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-.7
pgfIlogarithmic(.5,params)

 ## The function is currently defined as

pgfIlogarithmic <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
    stop("The length of params is 1")
    theta<-params[1]
if ((theta>=1)|(theta<=0))
    stop ("Parameter theta belongs to the interval (0,1)")
    (1-theta^s)/(1-theta)
}
```

---

pgfIlogarithmicbinomial

Function pgfIlogarithmicbinomial

Description

This function calculates value of the pgf's inverse of the logarithmic-binomial distribution.

Usage

```
pgfIlogarithmicbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the logarithmic-binomial distribution, such that params<-c(p1,p2,m,n), where p1, p2 are probabilities, and m, n are positive integers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.9,.7,4)
pgfIlogarithmicbinomial(.5,params)

## The function is currently defined as

pgfIlogarithmicbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
   warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
   stop("At least one value in params is missing")
if (length(params)>3)
   stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if ((theta>=1)|(theta<=0))
   stop ("Parameter theta belongs to the interval (0,1)")
if ((p>=1)|(p<=0))
   stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    zval<-(1-theta^s)/(1-theta)
    (zval^(1/n)-1+p)/p
}
```

`pgfIlogarithmicpoisson`

Function pgfIlogarithmicpoisson

Description

This function calculates value of the pgf's inverse of the logarithmic-Poisson distribution.

Usage

```
pgfIlogarithmicpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the logarithmic-Poisson distribution, such that params<-c(theta,lambda), where theta is probability, and lambda is the positive number.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.9,7)
pgfIlogarithmicpoisson(.5,params)

## The function is currently defined as
pgfIlogarithmicpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
     stop("At least one value in params is missing")
if (length(params)>2)
     stop("The length of params is 2")
     theta<-params[1]
    lambda<-params[2]
if ((theta>=1)|(theta<=0))
    stop ("Parameter theta belongs to the interval (0,1)")
if (lambda<=0)
    stop ("Parameter lambda must be positive")

    zval<-(1-theta^s)/(1-theta)
    1+log(zval)/lambda
}
```

---

pgfInegativebinomial *Function pgfInegativebinomial*

Description

This function calculates value of the pgf's inverse of the negative binomial distribution.

Usage

```
pgfInegativebinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the negative binomial distribution, such that params<-c(theta,k), where theta is probability, and k is positive number.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.9,.7)
pgfInegativebinomial(.5,params)


## The function is currently defined as

pgfDnegativebinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("Some  elements of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    k<-params[2]
if ((theta>=1)|(theta<=0))
    stop ("Parameter theta belongs to the interval (0,1)")
if (k<=0)
    stop("Parameter k must be positive")
    k*(1-theta)*theta^k/(1-(1-theta)*s)^(k+1)
}
```

pgfIneymantypea *Function pgfIneymantypea*

Description

This function calculates value of the pgf's inverse of the Neyman type A distribution.

Usage

```
pgfIneymantypea(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type A distribution, such that params<-c(theta,lambda), where both parameters are positive numbers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,5)
pgfIneymantypea(.5,params)


##The function is currently defined as

pgfIneymantypea <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
     stop("At least one value in params is missing")
if (length(params)>2)
     stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if (s<=exp(-lambda))
    stop("Logarithm function is not defined. ")
```

```
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")
    1+1/theta*log(1+log(s)/lambda)
}
```

---

pgfIneymantypeb *Function pgfIneymantypeb*

Description

This function calculates value of the pgf's inverse of the Neyman type B distribution.

Usage

```
pgfIneymantypeb(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type B distribution, such that params<-c(theta,lambda), where both parameters are positive numbers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,5)
pgfIneymantypeb(.9,params)


##The function is currently defined as

pgfIneymantypeb <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
```

```
        func<-function(x) pgfneymantypeb(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

---

pgfIneymantypec *Function pgfIneymantypec*

Description

This function calculates value of the pgf of the Neyman type C distribution.

Usage

```
pgfIneymantypec(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type C distribution, such that params<-c(theta,lambda), where both parameters are positive numbers

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,5)
pgfIneymantypec(.9,params)

## The function is currently defined as

pgfIneymantypec <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfneymantypec(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
```

```
    }
    xval
}
```

---

pgfIpascalpoisson *Function pgfIpascalpoisson*

Description

This function calculates value of the pgf's inverse of the Pascal Poisson distribution.

Usage

```
pgfIpascalpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be form interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Pascal Poisson distribution, such that params<-c(theta,mu, a), where all parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(3,2,.5)
pgfIpascalpoisson(.9,params)


## The function is currently defined as

pgfIpascalpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
    stop("The length of params is 3")
    theta<-params[1]
```

```
    mu<-params[2]
    a<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (mu<=0)
    stop ("Parameter mu must be positive")
if (a<=0)
    stop ("Parameter a must be positive")
    zval<-1+mu/(a*theta)-s^(-1/a)
    1+log(a*theta*zval/mu)/theta
}
```

---

pgfIpoisson *Function pgfIpoisson*

Description

This function calculates value of the pgf's inverse of the Poisson distribution.

Usage

```
pgfIpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Positive parameter of the Poisson distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-7
pgfIpoisson(.5,params)


## The function is currently defined as

pgfIpoisson <- function(s,params) {
k<-s[abs(s)>1]
```

```
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
 if (length(params)>1) stop("The length of params is 1")
    theta<-params[1]
if (theta<=0)
    stop ("Parameter of Poisson distribution must be positive")
    1+log(s)/theta
}
```

`pgfIpoissonbinomial` *Function pgfIpoissonbinomial*

Description

This function calculates value of the pgf's inverse of the Poisson-binomial distribution.

Usage

```
pgfIpoissonbinomial(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Poisson-binomial distribution, such that params<-c(theta,p,n), where theta is positive number, p is probability, and n is positive integer.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(2.6,.7,3)
pgfIpoissonbinomial(.5,params)


## The function is currently defined as

pgfIpoissonbinomial <- function(s,params) {
k<-s[abs(s)>1]
```

```
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
    stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    zval<-1+log(s)/theta
    (zval^(1/n)-1+p)/p
}
```

`pgfIpoissonlindley` *Function pgfIpoissonlindley*

Description

This function calculates value of the pgf's inverse derivative of the Poisson-Lindley distribution.

Usage

```
pgfIpoissonlindley(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	Positive parameter of the Poisson-Lindley distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-5
pgfIpoissonlindley(.9,params)


## The function is currently defined as

pgfIpoissonlindley <- function(s,params) {
    xval<-length(s)
    theta<-params[1]
    for (i in 1:length(s)) {
        func<-function(x) pgfpoissonlindley(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

`pgfIpoissonpascal` *Function pgfIpoissonpascal*

Description

This function calculates value of the pgf's inverse of the Poisson-Pascal distribution.

Usage

```
pgfIpoissonpascal(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Poisson-Pascal distribution, such that params<-c(theta,p,k), where all parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.4,.3)
pgfIpoissonpascal(.9,params)

## The function is currently defined as

pgfIpoissonpascal <- function(s,params) {
m<-s[abs(s)>1]
if (length(m)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
     stop("At least one value in params is missing")
if (length(params)>3)
     stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    k<-params[3]
if (theta<=0)
     stop ("Parameter theta must be positive")
if (p<=0)
     stop ("Parameter lambda must be positive")
if (k<=0)
     stop ("Parameter k must be positive")
    (1+p-(1+log(s)/theta)^(-1/k))/p
}
```

`pgfIpolyaaeppli` *Function pgfIpolyaaeppli*

Description

This function calculates value of the pgf's inverse of the Polya Aeppli distribution.

Usage

```
pgfIpolyaaeppli(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Polya Aeppli distribution, such that params<-c(theta,p), where theta is positive number, and p is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.4)
pgfIpolyaaeppli(.5,params)


## The function is currently defined as

pgfIpolyaaeppli <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
     stop("At least one value in params is missing")
if (length(params)>2)
     stop("The length of params is 2")
    theta<-params[1]
    p<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")

    (theta+log(s))/(theta+p*log(s))
}
```

`pgfIthomas` *Function pgfIthomas*

Description

This function calculates value of the pgf's inverse of the Thomas distribution.

Usage

```
pgfIthomas(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Thomas distribution, such that params<-c(lambda,theta), where both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.4)
pgfIthomas(.9,params)


## The function is currently defined as

pgfIthomas <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfthomas(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    print(xval)
    xval
}
```

---

pgfIwaring *Function pgfIwaring*

Description

This function calculates value of the pgf's inverse of the Waring distribution.

Usage

```
pgfIwaring(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Waring distribution, such that params<-c(c,a), where $c>a$ and both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.8,.4)
pgfIwaring(.9,params)

## The function is currently defined as

pgfIwaring <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfwaring(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

---

pgfIyule *Function pgfIyule*

Description

This function calculates value of the pgf's inverse of the Yule distribution.

Usage

```
pgfIyule(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Postive parameter of the Yule distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-.3
pgfIyule(.9,params)


## The function is currently defined as

pgfIyule <- function(s,params) {
    xval<-length(s)
    for (i in 1:length(s)) {
        func<-function(x) pgfyule(x,params)-s[i]
        xval[i]<-uniroot(func,lower=0,upper=1)$root
    }
    xval
}
```

`pgfkattitypeh1` *Function pgfkattitypeh1*

Description

This function calculates value of the pgf of the Katti type H1 distribution.

Usage

```
pgfkattitypeh1(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Katti type H1 distribution, such that params<-c(theta,a,b), where all parameter are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.2,4,.5)
pgfkattitypeh1(.5,params)


## The function is currently defined as

pgfkattitypeh1 <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
       warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
       stop("At least one value in params is missing")
if (length(params)>3)
       stop("The length of params is 3")
    theta<-params[1]
    a<-params[2]
    b<-params[3]
if (theta<=0)
       stop ("Parameter theta must be positive")
if (a<=0)
       stop ("Parameter a must be positive")
if (b<=0)
       stop ("Parameter b must be positive")

    genhypergeo(a,a+b,theta*(s-1))
}
```

---

pgfkattitypeh2 *Function pgfkattitypeh2*

Description

This function calculates value of the pgf of the Katti type H2 distribution.

Usage

```
pgfkattitypeh2(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Katti type H2 distribution, such that params<-c(theta,a,b,k), where all parameter are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.2,4,.5,.3)
pgfkattitypeh2(.5,params)


## The function is currently defined as

pgfkattitypeh2 <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
      warning("At least one element of the vector s are out of interval [-1,1]")

if (length(params)<4)
      stop("At least one value in params is missing")
if (length(params)>4)
      stop("The length of params is 4")
    theta<-params[1]
    a<-params[2]
    b<-params[3]
    k<-params[4]
if (theta<=0)
     stop ("Parameter theta must be positive")
if (a<=0)
     stop ("Parameter a must be positive")
if (b<=0)
     stop ("Parameter b must be positive")
if (k<=0)
     stop ("Parameter k must be positive")

    Re(hypergeo(k,a,a+b,theta*(s-1)))
```

```
}
```

---

pgflogarithmic *Function pgflogarithmic*

Description

This function calculates value of the pgf of the logarithmic distribution.

Usage

```
pgflogarithmic(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Paramemeter of logarithmic distribution, such that params<-theta, where theta is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-0.3
pgflogarithmic(.5,params)


## The function is currently defined as

pgflogarithmic <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
      warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
      stop("The length of params is 1")
      theta<-params[1]
if ((theta>=1)|(theta<=0))
stop ("Parameter theta belongs to the interval (0,1)")
    log(1-(1-theta)*s)/log(theta)
}
```

pgflogarithmicbinomial

Function pgflogarithmicbinomial

Description

This function calculates value of the pgf of the logarithmic-binomial distribution.

Usage

```
pgflogarithmicbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the logarithmic binomial distribution, such that params<-c(p1,p2,m,n), where p1, p2 are probabilities, and m, n are positive integers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.5,.3,5)
pgflogarithmicbinomial(.5,params)


## The function is currently defined as

pgflogarithmicbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
      warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
      stop("At least one value in params is missing")
if (length(params)>3)
      stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
```

```
if ((theta>=1)|(theta<=0))
      stop ("Parameter theta belongs to the interval (0,1)")
if ((p>=1)|(p<=0))
      stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
     log(1-(1-theta)*(1-p+p*s)^n)/log(theta)
}
```

---

pgflogarithmicpoisson *Function pgflogarithmicpoisson*

Description

This function calculates value of the pgf of the logarithmic-Poisson distribution.

Usage

```
pgflogarithmicpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the logarithmic-Poisson distribution, such that params<-c(theta,lambda), where theta is probability, and lambda is positive number.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.5,4)
pgflogarithmicpoisson(.5,params)

## The function is currently defined as

pgflogarithmicpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
```

```
      warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
      stop("At least one value in params is missing")
if (length(params)>2)
      stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if ((theta>=1)|(theta<=0))
       stop ("Parameter theta belongs to the interval (0,1)")
if (lambda<=0)
       stop ("Parameter lambda must be positive")
    log(1-(1-theta)*exp(lambda*(s-1)))/log(theta)
}
```

`pgfnegativebinomial` *Function pgfnegativebinomial*

Description

This function calculates value of the pgf of the negative binomial distribution.

Usage

```
pgfnegativebinomial(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the negative binomial distribution, such that params<-c(theta,k), where theta is probability, and k is positive number.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.2,.3)
pgfnegativebinomial(.5,params)


## The function is currently defined as

pgfnegativebinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("Some  elements of the vector s are out of interval [-1,1]")
if (length(params)<2)
     stop("At least one value in params is missing")
if (length(params)>2)
     stop("The length of params is 2")
    theta<-params[1]
    k<-params[2]
if ((theta>=1)|(theta<=0))
     stop ("Parameter theta belongs to the interval (0,1)")
if (k<=0)
     stop("Parameter k must be positive")
    (theta/(1-(1-theta)*s))^k
}
```

---

pgfneymantypea *Function pgfneymantypea*

Description

This function calculates value of the pgf of the Neyman type A distribution.

Usage

```
pgfneymantypea(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type A distribution, such that params<-c(theta,lambda), where both parameters are positive numbers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.6)
pgfneymantypea(.5,params)

## The function is currently defined as

pgfneymantypea<-function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
     stop("At least one value in params is missing")
if (length(params)>2)
     stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")
     exp(lambda*(exp(theta*(s-1))-1))
}
```

---

pgfneymantypeb *Function pgfneymantypeb*

Description

This function calculates value of the pgf of the Neyman type B distribution.

Usage

```
pgfneymantypeb(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type B distribution, such that params<-c(theta,lambda), where both parameters are positive numbers.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.6)
pgfneymantypeb(.5,params)

## The function is currently defined as

pgfneymantypeb <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (lambda<=0)
    stop ("Parameter lambda must be positive")

    exp(lambda*(genhypergeo(1,2,theta*(s-1))-1))
}
```

`pgfneymantypec` *Function pgfneymantypec*

Description

This function calculates value of the pgf of the Neyman type C distribution.

Usage

```
pgfneymantypec(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Neyman type C distribution, such that params<-c(theta,lambda), where both parameters are positive numbers

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.6)
pgfneymantypec(.5,params)


##The function is currently defined as

pgfneymantypec <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
       stop("At least one value in params is missing")
if (length(params)>2)
       stop("The length of params is 2")
    theta<-params[1]
    lambda<-params[2]
if (theta<=0)
       stop ("Parameter theta must be positive")
if (lambda<=0)
       stop ("Parameter lambda must be positive")
    exp(lambda*(genhypergeo(1,3,theta*(s-1))-1))
}
```

pgfpascalpoisson *Function pgfpascalpoisson*

Description

This function calculates value of the pgf of the Pascal Poisson distribution.

Usage

```
pgfpascalpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Pascal Poisson distribution, such that params<-c(theta,mu, a), where all parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.6,4)
pgfpascalpoisson(.5,params)


##The function is currently defined as

pgfpascalpoisson <- function(s,params) {
    k<-s[abs(s)>1]
if (length(k)>0)
     warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
     stop("At least one value in params is missing")
if (length(params)>3)
     stop("The length of params is 3")
    theta<-params[1]
    mu<-params[2]
    a<-params[3]
if (theta<=0)
```

```
    stop ("Parameter theta must be positive")
if (mu<=0)
    stop ("Parameter mu must be positive")
if (a<=0)
    stop ("Parameter a must be positive")
    (1+mu/(a*theta)-mu/(a*theta)*exp(theta*(s-1)))^(-a)
}
```

---

pgfpoisson *Function pgfpoisson*

Description

This function calculates value of the pgf of the Poisson distribution.

Usage

```
pgfpoisson(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Positive parameter of the Poisson distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-5
pgfpoisson(.2,params)

## The function is currently defined as

pgfpoisson <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (missing(params))
    stop("Distribution parameters are not defined")
    theta<-params[1]
```

```
if (theta<=0)
    stop ("Parameter of Poisson distribution must be positive")
    exp(theta*(s-1))
}
```

---

pgfpoissonbinomial *Function pgfpoissonbinomial*

Description

This function calculates value of the pgf of the Poisson-binomial distribution.

Usage

```
pgfpoissonbinomial(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Poisson-binomial distribution, such that params<-c(theta,p,n), where theta is positive number, p is probability, and n is positive integer.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(4,.5,2)
pgfpoissonbinomial(.5,params)

##The function is currently defined as

pgfpoissonbinomial <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
```

```
    stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    n<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")
if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")
    exp(theta*((1-p+p*s)^n-1))
}
```

---

pgfpoissonlindley *Function pgfpoissonlindley*

Description

This function calculates value of the pgf of the Poisson-Lindley distribution.

Usage

```
pgfpoissonlindley(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	Positive parameter of the Poisson-Lindley distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-4
pgfpoissonlindley(.5,params)


## The function is currently defined as

pgfpoissonlindley <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
    stop("The length of params is 1")
theta<-params[1]
if (theta<=0)
    stop ("Parameter lambda must be positive")
theta^2*(theta+2-s)/((theta+1)*(theta+1-s)^2)
}
```

---

pgfpoissonpascal *Function pgfpoissonpascal*

Description

This function calculates value of the pgf of the Poisson-Pascal distribution.

Usage

```
pgfpoissonpascal(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Poisson-Pascal distribution, such that params<-c(theta,p,k), where all parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.6,4)
pgfpoissonpascal(.5,params)


## The function is currently defined as

pgfpoissonpascal <- function(s,params) {
m<-s[abs(s)>1]
if (length(m)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<3)
    stop("At least one value in params is missing")
if (length(params)>3)
    stop("The length of params is 3")
    theta<-params[1]
    p<-params[2]
    k<-params[3]
if (theta<=0)
    stop ("Parameter theta must be positive")
if (p<=0)
    stop ("Parameter lambda must be positive")
if (k<=0)
    stop ("Parameter k must be positive")
exp(theta*((1+p-p*s)^(-k)-1))
}
```

`pgfpolyaaeppli` *Function pgfpolyaaeppli*

Description

This function calculates value of the pgf of the Polya Aeppli distribution.

Usage

```
pgfpolyaaeppli(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Polya Aeppli distribution, such that params<-c(theta,p), where theta is positive number, and p is probability.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(5,.6)
pgfpolyaaeppli(.5,params)



## The function is currently defined as

pgfpolyaaeppli <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    theta<-params[1]
    p<-params[2]
if (theta<=0)
    stop ("Parameter theta must be positive")
if ((p>=1)|(p<=0))
    stop ("Parameter p belongs to the interval (0,1)")
exp(theta/p*((1-p)/(1-p*s)-1))
}
```

`pgfthomas` *Function pgfthomas*

Description

This function calculates value of the pgf of the Thomas distribution.

Usage

```
pgfthomas(s, params)
```

Arguments

`s`	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf diverges.
`params`	List of the parameters of the Thomas distribution, such that params<-c(lambda,theta), where both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(.4,2)
pgfthomas(.5,params)

## The function is currently defined as

pgfthomas <- function(s,params) {
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    lambda<-params[1]
    theta<-params[2]
if (lambda<=0)
    stop ("Parameter lambda must be positive")
if (theta<=0)
    stop ("Parameter theta must be positive")
exp(lambda*(s*exp(theta*(s-1))-1))
}
```

---

pgfwaring *Function pgfwaring*

Description

This function calculates value of the pgf of the Waring distribution.

Usage

```
pgfwaring(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be form interval [-1,1]. In the opposite pgf diverges.
params	List of the parameters of the Waring distribution, such that params<-c(c,a), where $c>a$ and both parameters are positive.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-c(3,2)
pgfwaring(.5,params)

## The function is currently defined as

pgfwaring <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)<2)
    stop("At least one value in params is missing")
if (length(params)>2)
    stop("The length of params is 2")
    cc<-params[1]
    a<-params[2]
if (cc<=0)
    stop ("Parameter c must be positive")
if (a<=0)
    stop ("Parameter a must be positive")
(cc-a)/cc*Re(hypergeo(1,a,cc+1,s))
}
```

pgfyule *Function pgfyule*

Description

This function calculates value of the pgf of the Yule distribution.

Usage

```
pgfyule(s, params)
```

Arguments

s	Value of the parameter of the pgf. It should be from interval [-1,1]. In the opposite pgf divegrates.
params	Postive parameter of the Yule distribution, such that params<-theta.

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Johnson N, Kotz S, Kemp A (1992) Univariate Discrete Distributions, John Wiley and Sons, New York

Hankin R.K.S, Lee A (2006) A new family of non-negative distributions. Australia and New Zealand Journal of Statistics 48(1): 67(78)

http://www.am.qub.ac.uk/users/g.gribakin/sor/Chap3.pdf

Examples

```
params<-3
pgfyule(.5,params)

## The function is currently defined as

pgfyule <- function(s,params) {
    require(hypergeo)
k<-s[abs(s)>1]
if (length(k)>0)
    warning("At least one element of the vector s are out of interval [-1,1]")
if (length(params)>1)
    stop("The length of params is 1")
    theta<-params[1]
if (theta<=0)
    stop ("Parameter theta must be positive")
theta/(theta+1)*Re(hypergeo(1,1,theta+2,s))
}
```

qCompound *function qCompound*

Description

Function qCompound calculates quantiles of the random variable X.

Usage

```
qCompound(p,parent,compound,compoundDist,params,...)
```

Arguments

p	vector of probabilities
parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See

http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Nadarajah S, Popovic B.V, Ristic M.M (2011) Compounding: An R Package for Computing Continuous Distributions Obtained by Compounding a Continuous and a Discrete Distribution (submitted)

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
```

```
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
p<-0.5
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
qCompound(p,parentD,compoundD,params,shape1=2,shape2=0.3)

## The function is currently defined as
qCompound <- function(p,parent,compound,compoundDist,params,...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }

    l<-p[p<0|p>1]
    if (length(l)>0) stop("Parameter p is probability")

    xval <- real(length(p))
    Finv <- get(paste("q", parent, sep = ""), mode = "function")
    phi <- get(paste("pgf", compound, sep = ""), mode = "function")
    phiInv <- get(paste("pgfI", compound, sep = ""), mode = "function")
    xval <- Finv(1-phiInv(1-p*(1-phi(0,params)),params),...)
    return(xval)
}
```

---

rCompound *function rCompound*

Description

Function rCompound generates sample for the random variable X.

Usage

```
rCompound(n,parent,compound,compoundDist,params,...)
```

Arguments

n	number of observations
parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions

params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See

http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Nadarajah S, Popovic B.V, Ristic M.M (2011) Compounding: An R Package for Computing Continuous Distributions Obtained by Compounding a Continuous and a Discrete Distribution (submitted)

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
n<-5
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
rCompound(n,parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as
rCompound <- function(n, parent, compound,params, ...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
    if (n<0)
     stop("Parameter n must be positive")
 if(!(abs(n-round(n))<.Machine$double.eps^0.5))
stop("Parameter n must be positive integer")

    zval <- runif(n)
    xval <- qCompound(zval,parent,compound,compoundDist,params,...)
```

```
    return(xval)
}
```

---

skewCompound *function skewCompound*

Description

Function skewCompound calculates skewness of the random variable X.

Usage

```
skewCompound(parent,compound,compoundDist,params,...)
```

Arguments

parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.
...	Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See

http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Nadarajah S, Popovic B.V, Ristic M.M (2011) Compounding: An R Package for Computing Continuous Distributions Obtained by Compounding a Continuous and a Discrete Distribution (submitted)

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
skewCompound(parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as
skewCompound <- function(parent,compound,params,...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
   m1 <- meanCompound(parent,compound,compoundDist,params,...)
   m3 <- momentCompound(3,parent,compound,compoundDist,params,...)
   sig2 <- varCompound(parent,compound,compoundDist,params,...)
   return((m3-3*m1*sig2-m1^3)/sig2^(3/2))
}
```

---

varCompound *function varCompound*

Description

Function varCompound calculates variance of the random variable X.

Usage

```
varCompound(parent,compound,compoundDist,params,...)
```

Arguments

parent	name of the parent distribution. It can be any continuous distribution supported by R.
compound	name of the compound distribution. It can be any discrete distribution supported by this package.
compoundDist	list of available compounding distributions
params	Parameter or list of parameters of compounding distribution.

... Parameters of continuous distribution could be provided as additional parameters.

Details

Parameters of the parent distribution must be provided in the same way as it is in built in R functions. See

http://127.0.0.1:23174/library/stats/html/Distributions.html

Author(s)

S. Nadarajah, B. V. Popovic, M. M. Ristic

References

Nadarajah S, Popovic B.V, Ristic M.M (2011) Compounding: An R Package for Computing Continuous Distributions Obtained by Compounding a Continuous and a Discrete Distribution (submitted)

Examples

```
compoundDist <- c("geometric","poisson","negativebinomial","binomial",
"logarithmic","binomialbinomial","binomialpoisson",
"poissonbinomial","neymantypea","polyaaeppli",
"poissonpascal","pascalpoisson",
"logarithmicbinomial","logarithmicpoisson",
"poissonlindley",
"hyperpoisson","yule","waring","kattitypeh1",
"kattitypeh2","neymantypeb","neymantypec",
"hypergeometric","thomas")
parentD<-"beta"
compoundD<-"hypergeometric"
params<-c(3,2,0.5)
varCompound(parentD,compoundD,compoundDist,params,shape1=2,shape2=0.3)

## The function is currently defined as
varCompound <- function(parent,compound,compoundDist,params,...) {
    if (!exists(paste("p",parent,sep=""))) {
        return(paste("The parent distribution",parent,"doesn't exist"))
    }
    if (!is.element(compound,compoundDist)) {
        return(paste("The discrete distribution",compound,"doesn't exist"))
    }
    m1 <- momentCompound(1,parent,compound,compoundDist,params,...)
    m2 <- momentCompound(2,parent,compound,compoundDist,params,...)
    return(m2-m1^2)
}
```

Index