

Package ‘polyaAeppli’

July 23, 2025

Type Package

Title Implementation of the Polya-Aeppli Distribution

Version 2.0.2

Depends R ($\geq 3.0.0$)

Date 2022-04-21

Author Conrad Burden

Maintainer Conrad Burden <conrad.burden@anu.edu.au>

Description Functions for evaluating the mass density, cumulative distribution function, quantile function and random variate generation for the Polya-Aeppli distribution, also known as the geometric compound Poisson distribution. More information on the implementation can be found at Conrad J. Burden (2014) <[doi:10.48550/arXiv.1406.2780](https://doi.org/10.48550/arXiv.1406.2780)>.

License GPL (≥ 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2022-04-21 11:10:04 UTC

Contents

polyaAeppli-package	1
PolyaAeppli	3
Index	5

polyaAeppli-package	<i>Implementation of the Polya-Aeppli Distribution</i>
---------------------	--

Description

Functions for evaluating the mass density, cumulative distribution function, quantile function and random variate generation for the Polya-Aeppli distribution, also known as the geometric compound Poisson distribution.
More information on the implementation of **polyaAeppli** can be found at Conrad J. Burden (2014) <[arXiv:1406.2780](https://arxiv.org/abs/1406.2780)>.

Details

```

Package:  polyaAeppli
Type:     Package
Version:  2.0.2
Depends:  R (>= 3.0.0)
Date:     2020-04-21
License:  GPL(>=2)

```

Consistent with the conventions used in R package stats, this implementation of the Polya-Aeppli distribution comprises the four functions

```

dPolyaAeppli(x, lambda, prob, log = FALSE)
pPolyaAeppli(q, lambda, prob, lower.tail = TRUE, log.p = FALSE)
qPolyaAeppli(p, lambda, prob, lower.tail = TRUE, log.p = FALSE)
rPolyaAeppli(n, lambda, prob)

```

Author(s)

Conrad Burden

Maintainer: conrad.burden@anu.edu.au

References

Johnson NL, Kotz S, Kemp AW (1992). *Univariate Discrete Distributions*. 2nd edition. Wiley, New York.

Nuel G (2008). *Cumulative distribution function of a geometric Poisson distribution*. Journal of Statistical Computation and Simulation, **78**(3), 385-394.

Examples

```

lambda <- 8
prob <- 0.2
## Plot histogram of random sample
PASample <- rPolyaAeppli(10000, lambda, prob)
maxPA <- max(PASample)
hist(PASample, breaks=(0:(maxPA + 1)) - 0.5, freq=FALSE,
      xlab = "x", ylab = expression(P[X](x)), main="", border="blue")
## Add plot of density function
x <- 0:maxPA
points(x, dPolyaAeppli(x, lambda, prob), type="h", lwd=2)

lambda <- 4000
prob <- 0.005
qq <- 0:10000
## Plot log of the extreme lower tail p-value
log.pp <- pPolyaAeppli(qq, lambda, prob, log.p=TRUE)
plot(qq, log.pp, type = "l", ylim=c(-lambda,0),

```

```

xlab = "x", ylab = expression("log Pr(X " <= "x)"))
## Plot log of the extreme upper tail p-value
log.1minuspp <- pPolyaAeppli(qq, lambda, prob, log.p=TRUE, lower.tail=FALSE)
points(qq, log.1minuspp, type = "l", col = "red")
legend("topright", c("lower tail", "upper tail"),
col=c("black", "red"), lty=1, bg="white")

```

PolyaAeppli

Polya-Aeppli

Description

Density, distribution function, quantile function and random generation for the Polya-Aeppli distribution with parameters `lambda` and `prob`.

Usage

```

dPolyaAeppli(x, lambda, prob, log = FALSE)
pPolyaAeppli(q, lambda, prob, lower.tail = TRUE, log.p = FALSE)
qPolyaAeppli(p, lambda, prob, lower.tail = TRUE, log.p = FALSE)
rPolyaAeppli(n, lambda, prob)

```

Arguments

<code>x</code>	vector of quantiles
<code>q</code>	vector of quantiles
<code>p</code>	vector of probabilities
<code>n</code>	number of random variables to return
<code>lambda</code>	a vector of non-negative Poisson parameters
<code>prob</code>	a vector of geometric parameters between 0 and 1
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise $P[X > x]$

Details

A Polya-Aeppli, or geometric compound Poisson, random variable is the sum of a Poisson number of identically and independently distributed shifted geometric random variables. Its distribution (with `lambda`= λ , `prob`= p) has density

$$Prob(X = x) = e^{-(\lambda - \lambda p)} \frac{\lambda^x}{x!} (1-p)^x$$

for $x = 0$;

$$Prob(X = x) = e^{-(\lambda - \lambda p)} \sum_{n=0}^x \frac{\lambda^n}{n!} \frac{x!}{(x-n)!} p^{x-n} (1-p)^n$$

for $x = 1, 2, \dots$

If an element of x is not integer, the result of `dPolyaAeppli` is zero, with a warning.

The quantile is right continuous: `qPolyaAeppli(p, lambda, prob)` is the smallest integer x such that $P(X \leq x) \geq p$.

Setting `lower.tail = FALSE` enables much more precise results when the default, `lower.tail = TRUE` would return 1, see the example below.

Value

`dPolyaAeppli` gives the (log) density, `pPolyaAeppli` gives the (log) distribution function, `qPolyaAeppli` gives the quantile function, and `rPolyaAeppli` generates random deviates.

Invalid `lambda` or `prob` will terminate with an error message.

Author(s)

Conrad Burden

References

Johnson NL, Kotz S, Kemp AW (1992). *Univariate Discrete Distributions*. 2nd edition. Wiley, New York.

Nuel G (2008). *Cumulative distribution function of a geometric Poisson distribution*. Journal of Statistical Computation and Simulation, **78**(3), 385-394.

Examples

```
lambda <- 8
prob <- 0.2
## Plot histogram of random sample
PAsample <- rPolyaAeppli(10000, lambda, prob)
maxPA <- max(PAsample)
hist(PAsample, breaks=(0:(maxPA + 1)) - 0.5, freq=FALSE,
     xlab = "x", ylab = expression(P[X](x)), main="", border="blue")
## Add plot of density function
x <- 0:maxPA
points(x, dPolyaAeppli(x, lambda, prob), type="h", lwd=2)

lambda <- 4000
prob <- 0.005
qq <- 0:10000
## Plot log of the extreme lower tail p-value
log.pp <- pPolyaAeppli(qq, lambda, prob, log.p=TRUE)
plot(qq, log.pp, type = "l", ylim=c(-lambda,0),
     xlab = "x", ylab = expression("log Pr(X " <= "x)"))
## Plot log of the extreme upper tail p-value
log.1minuspp <- pPolyaAeppli(qq, lambda, prob, log.p=TRUE, lower.tail=FALSE)
points(qq, log.1minuspp, type = "l", col = "red")
legend("topright", c("lower tail", "upper tail"),
     col=c("black", "red"), lty=1, bg="white")
```

Index

dPolyaAeppli (PolyaAeppli), [3](#)

PolyaAeppli, [3](#)

polyaAeppli (polyaAeppli-package), [1](#)

polyaAeppli-package, [1](#)

pPolyaAeppli (PolyaAeppli), [3](#)

qPolyaAeppli (PolyaAeppli), [3](#)

rPolyaAeppli (PolyaAeppli), [3](#)