The Skellam Distribution

Description
Density, distribution function, quantile function and random number generation for the Skellam distribution.

Usage
- `dskellam(x, lambda1, lambda2 = lambda1, log = FALSE)`
- `pskellam(q, lambda1, lambda2 = lambda1, lower.tail = TRUE, log.p = FALSE)`
- `qskellam(p, lambda1, lambda2 = lambda1, lower.tail = TRUE, log.p = FALSE)`
- `rskellam(n, lambda1, lambda2 = lambda1)`
- `dskellam.sp(x, lambda1, lambda2 = lambda1, log = FALSE)`
- `pskellam.sp(q, lambda1, lambda2 = lambda1, lower.tail = TRUE, log.p = FALSE)`

Arguments
- `x, q` vector of quantiles.
- `p` vector of probabilities.
- `n` number of observations. If `length(n) > 1`, the length is taken to be the number required.
- `lambda1, lambda2` vectors of (non-negative) means.
- `log, log.p` logical; if `TRUE`, probabilities `p` are given as `log(p)`.
- `lower.tail` logical; if `TRUE` (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details
If $Y_1$ and $Y_2$ are Poisson variables with means $\mu_1$ and $\mu_2$ and correlation $\rho$, then $X = Y_1 - Y_2$ is Skellam with parameters $\lambda_1 = \mu_1 - \rho \sqrt{\mu_1 \mu_2}$ and $\lambda_2 = \mu_2 - \rho \sqrt{\mu_1 \mu_2}$.

dskellam returns a value equivalent to

\[ I(2 \sqrt{\lambda_1 \lambda_2}, |x|)(\lambda_1 / \lambda_2)^{x/2} \exp(-\lambda_1 - \lambda_2) \]

where $I(y, \nu)$ is the modified Bessel function of the first kind. The $|x|$ differs from most Skellam expressions in the literature, but is correct since $x$ is an integer, resulting in improved portability and (in R versions prior to 2.9) improved accuracy for $x<0$. Exponential scaling is used in besselI to postpone numerical problems. When numerical problems do occur, a saddlepoint approximation is substituted, which typically gives at least 4-figure accuracy. An alternative representation is `dchisq(2\lambda_1, 2(x + 1), 2\lambda_2)\lambda_2$ for $x \geq 0$, and `dchisq(2\lambda_2, 2(1 - x), 2\lambda_1)\lambda_1$ for $x \leq 0$; but in R `besselI` appears to be more accurately implemented (for very small probabilities) than `dchisq`.
pskellam(x, lambda1, lambda2) returns pchisq(2*lambda2, -2*x, 2*lambda1) for x ≤ 0 and 1 - pchisq(2*lambda1, 2*(x+1), 2*lambda2) for x > 0. When pchisq incorrectly returns 0, a saddlepoint approximation is substituted, which typically gives at least 2-figure accuracy.

The quantile is defined as the smallest value x such that F(x) ≥ p, where F is the distribution function. For lower.tail=FALSE, the quantile is defined as the largest value x such that F(x, lower.tail=FALSE) ≤ p.

dskellam and pskellam return saddlepoint approximations to the pmf and cdf. They are called by dskellam and pskellam when results from primary methods are in doubt.

Value

dskellam gives the (log) density, pskellam gives the (log) distribution function, qskellam gives the quantile function, and rskellam generates random deviates. Invalid lambdas will result in return value NaN, with a warning.

Note

The VGAM package also contains Skellam functions, which are syntactically similar; independently developed versions are included here for completeness. Moreover, this dskellam function offers a broader working range, correct handling of cases where at least one rate parameter is zero, enhanced argument checking, and (in R versions prior to 2.9) improved accuracy for x<0. If both packages are loaded, skellam::dskellam or vgam::dskellam can unambiguously specify which implementation to use.

Author(s)

Jerry W. Lewis, Patrick E. Brown

Source

The relation of dgamma to the modified Bessel function of the first kind was given by Skellam (1946). The relation of pgamma to the noncentral chi-square was given by Johnson (1959). Tables are given by Strackee and van der Gon (1962), which can be used to verify this implementation (cf. direct calculation in the examples below).

qskellam uses the Cornish–Fisher expansion to include skewness and kurtosis corrections to a normal approximation, followed by a search. If getOption("verbose")=TRUE, then qskellam will not use qpois when one of the lambdas is zero, in order to verify that this search algorithm has been implemented properly.

References


Examples

```r
require('skellam')

# one lambda = \( \lambda \) - Poisson
c(dskellam(0:10,5,0), dpois(0:10,5))
c(dskellam(-0:10),0,5), dpois(0:10,5))
c(pskellam(0:10,5,0,lower.tail=TRUE),
  ppois(0:10,5,lower.tail=TRUE))
c(pskellam(0:10,5,0,lower.tail=FALSE),
  ppois(0:10,5,lower.tail=FALSE))
c(pskellam(-0:10),0,5,lower.tail=FALSE),
  ppois(0:10-1,5,lower.tail=FALSE))
c(pskellam(-0:10),0,5,lower.tail=TRUE),
  ppois(0:10-1,5,lower.tail=FALSE))

# both lambdas \( \lambda \) - convolution of Poissons
dskellam(1,0.5,0.75) # sum(dpois(1+0:10,0.5)*dpois(0:10,0.75))
pskellam(1,0.5,0.75) # sum(dskellam(-10:1,0.5,0.75))
dskellam(c(-1),c(12,8),c(10,12)) # c(0.0697968,0.0697968)
dskellam(c(-1),c(12,8),c(10,12),log=TRUE)
  # log(dskellam(c(-1),c(12,8),c(10,12)))
dskellam(256,257,1)
  # 0.024829348733183769

# exact result for comparison with saddlepoint
dskellam(-3724,2000,3000)
  # 3.1058145363400105e-308

# exact result for comparison with saddlepoint
# (still accurate in extreme tail)
pskellam(c(-1,0),c(12,8),c(10,12)) # c(0.2965079,0.7034921)
pskellam(c(-1,0),c(12,8),c(10,12),lower.tail=FALSE)
  # 1-pskellam(c(-1,0),c(12,8),c(10,12))
pskellam(-2:2,8.5,10.25,log.p=TRUE) # log(pskellam(-2:2,8.5,10.25))
qskellam(c(0.05,0.95),3,4) # c(-5,3); pskellam(cbind(-6:-5,2:3),3,4)
qskellam(c(0.05,0.95),3,0) # c(1,6); qpois(c(0.05,0.95),3)
rskellam(35,8.5,10.25)
```

skellam.mle  

**MLE of the Skellam distribution**

Description

MLE of the Skellam distribution.
Usage

skellam.mle(x)

Arguments

x A vector of integers, positive or negative.

Details

Instead of having to maximise the log-likelihood with respect to the two parameters, \( \lambda_1 \) and \( \lambda_2 \), we maximise with respect to \( \lambda_2 \) and then \( \lambda_1 = \lambda_2 + \bar{x} \). This makes it faster. The command "nlm" is used to optimise the log-likelihood as it proved to be faster than the "optimise".

Value

A list including:

- iters The number of iterations required by "nlm".
- loglik The maximised log-likelihood value.
- param The estimated parameters, \( \hat{\lambda}_1 \) and \( \hat{\lambda}_2 \).

Author(s)

Michail Tsagris

References


Examples

```r
require('skellam')

x1 <- rpois(1000, 10)
x2 <- rpois(1000, 6)
x <- x1 - x2
skellam.mle(x)

x1 <- rpois(10000, 10)
x2 <- rpois(10000, 6)
x <- x1 - x2
skellam.mle(x)
```

Description

Regression assuming a Skellam distribution.

Usage

```r
skellam.reg(y, x)
```

Arguments

- `y`: A vector of integers, positive or negative.
- `x`: A matrix, a vector or a data.frame with the covariates.

Details

We use the exponential link function to ensure that the both $\lambda$s are positive. The command `rlm` does the main job.

Value

A list including:

- `loglik`: The maximised log-likelihood value.
- `param1`: The estimated regression coefficients of $\lambda_1$. This is matrix, with the first column being the estimated regression coefficients. The second column is their relevant standard error. The third column is the t value (coef/se(coef)) and the final column is the p-value of the Wald test.
- `param2`: The estimated regression coefficients of $\lambda_2$. This is matrix, with the first column being the estimated regression coefficients. The second column is their relevant standard error. The third column is the t value (coef/se(coef)) and the final column is the p-value of the Wald test.
Author(s)
Michail Tsagris

References


Examples
```
require('skellam')
set.seed(0)

x <- rnorm(1000)
y1 <- rpois(1000, exp(1 + 1 * x))
y2 <- rpois(1000, exp(-1 + 1 * x))
y <- y2 - y1
skellam.reg(y, x)
```
Index

*Topic **Maximum likelihood estimation**
  skellam.mle, 4

*Topic **Skellam regression**
  skellam.reg, 6

*Topic **distribution**
  Skellam, 2
  
  dskellam (Skellam), 2

  nlm, 6

  pskellam (Skellam), 2

  qskellam (Skellam), 2

  rskellam (Skellam), 2

  Skellam, 2, 3
  skellam (Skellam), 2
  skellam.mle, 4
  skellam.reg, 6