Package ‘GIGrvg’

March 22, 2023

Type Package
Title Random Variate Generator for the GIG Distribution
Version 0.8
Date 2023-03-22
Author Josef Leydold and Wolfgang Hormann
Maintainer Josef Leydold <josef.leydold@wu.ac.at>
Description Generator and density function for the Generalized Inverse Gaussian (GIG) distribution.
License GPL (>= 2)
LazyLoad yes
NeedsCompilation yes
Repository CRAN
Date/Publication 2023-03-22 14:00:08 UTC

R topics documented:

GIGrvg-package ......................................................... 1
rgig ................................................................. 3

Index

GIGrvg-package Generator and density for the Generalized Inverse Gaussian (GIG) distribution

Description

This package provides a generator and the density for the Generalized Inverse Gaussian (GIG) distribution. It uses the parametrization with density proportional to

\[ f(x) = \lambda^{-1} e^{-\frac{1}{2}(\chi/x + \psi x)} \]

Details
Package `GIGrvg` provides two routines:

- `rgig` generates GIG distributed random variates. It is especially designed for the varying parameter case, i.e., for sample size \( n=1 \).
- `dgig` computes the density of the GIG distribution.

**Note** that the parameters of the distribution are assumed to be single values. If a vector is provided then just the first value is used!

For the very fast generation of large samples more efficient algorithms exists. We recommend package `Runuran`.

**Author(s)**

Josef Leydold <josef.leydold@wu.ac.at> and Wolfgang Hörmann.

**References**


**Examples**

```r
## Draw a random sample
grig(n=10, lambda=0.5, chi=0.1, psi=2)

## Evaluate the density
dgig(0.3, lambda=0.5, chi=0.1, psi=2)
```
**rgig**

*Generator and Density of Generalized Inverse Gaussian (GIG) distribution.*

**Description**

Random variate generator for the Generalized Inverse Gaussian (GIG) distribution. The generator is especially designed for the varying parameter case, i.e., for sample size \( n=1 \).

**Usage**

```r
gig(n=1, lambda, chi, psi)

dgig(x, lambda, chi, psi, log = FALSE)
```

**Arguments**

- `n` Number of observations
- `lambda` Shape parameter
- `chi` Shape and scale parameter. Must be nonnegative for positive `lambda` and positive else.
- `psi` Shape and scale parameter. Must be nonnegative for negative `lambda` and positive else.
- `x` Argument of pdf
- `log` If `TRUE` the logarithm of the density will be returned.

**Details**

The package uses a parametrization for the GIG distribution where the density is proportional to

\[
 f(x) = x^{\lambda - 1}e^{-\frac{1}{2}(\chi/x + \psi x)}.
\]

The parameters have to satisfy the conditions

\[
 \lambda > 0, \psi > 0, \chi \geq 0, \quad \text{or}
\]

\[
 \lambda = 0, \psi > 0, \chi > 0, \quad \text{or}
\]

\[
 \lambda < 0, \psi \geq 0, \chi > 0.
\]

The generator is especially designed for the varying parameter case, i.e., for sample size \( n=1 \).

**Note** that the arguments `n`, `lambda`, `chi`, `psi` for these two R routines are assumed to be single values. If a vector is provided, then just the first value is used!

For the generation of large samples more efficient algorithms exist. We recommend package **Runuran**. The fast numeric inversion function `pinvd.new` is usable for GIG. It is about three times faster than `rgig` for large values of \( n \). However, it requires a slow set-up and is therefore not useful for the varying parameter case. For the usage of the Runuran functions see the last example below.
Routine *rgig* applies three different algorithms depending on the given parameters. When the density is $T$-concave (roughly spoken when $\lambda \geq 1$ or $\psi \chi \geq 1/4$) two variants of the Ratio-of-Uniforms method due to Lehner (1989) are used. These are quite similar to the widely used algorithm by Dagpunar but have a faster setup. When the density is not $T$-concave then a new algorithm with a uniformly rejection constant is used. (In the latter case Dagpunar’s algorithm may become extremely slow or may sample from an invalid distribution.)

**Value**

*rgig* creates a random sample of size $n$. In case of invalid arguments the routine simply stops execution.

*dgig* evaluates the density of the GIG distribution.

**Author(s)**

Josef Leydold <josef.leydold@wu.ac.at> and Wolfgang Hörmann.

**References**


**Examples**

```r
## Draw a random sample
x <- rgig(n=10, lambda=0.5, chi=0.1, psi=2)

## Evaluate the density
x <- dgig(0.3, lambda=0.5, chi=0.1, psi=2)

## Create a random sample and create a histogram
y <- rgig(n=10^5, 0.1, 2, 3)
hist(y, breaks=100, freq=FALSE)
xval <- seq(0, max(y), 0.01) # to add plot the corresponding density
lines(xval, dgig(xval, 0.1, 2, 3))

## Not run:
## Use a fast method from package Runuran for large samples
## (method PINV implements an approximate inversion method)
library("Runuran")
gen <- pinvd.new(udgig(theta=0.2, psi=0.05, chi=0.05))
x <- ur(gen, 10^6)
## End(Not run)
```
Index

* datagen
  GIGrvg-package, 1
  rgig, 3

* distribution
  GIGrvg-package, 1
  rgig, 3

* package
  GIGrvg-package, 1
  rgig, 3

dgig, 2
dgig(rgig), 3

GIGrvg (GIGrvg-package), 1
GIGrvg-package, 1

rgig, 2, 3
Runuran, 2, 3