Package ‘ginormal’

August 28, 2023

Type Package

Title Generalized Inverse Normal Distribution Density and Generation

Version 0.0.2

Description Density function and generation of random variables from the Generalized Inverse Normal (GIN) distribution from Robert (1991) <doi:10.1016/0167-7152(91)90174-P>. Also provides density functions and generation from the GIN distribution truncated to positive or negative reals. Theoretical guarantees supporting the sampling algorithms and an application to Bayesian estimation of network formation models can be found in the working paper Ding, Estrada and Montoya-Blandón (2023) <https://www.smontoyablandon.com/publication/networks/network_externalities.pdf>.

URL https://github.com/smonto2/ginormal

BugReports https://github.com/smonto2/ginormal/issues

License GPL (>= 3)

Depends R (>= 3.5.0)

Encoding UTF-8

Imports BAS

Suggests

RoxygenNote 7.2.3

NeedsCompilation yes

Author Santiago Montoya-Blandón [cre, aut] (<https://orcid.org/0000-0003-2229-2254>), Cheng Ding [aut], Juan Estrada [aut] (<https://orcid.org/0000-0003-0070-7586>), Zhilang Xia [aut], Shanjie Zhang [ctb], Jianming Jin [ctb]

Maintainer Santiago Montoya-Blandón <Santiago.Montoya-Blandon@glasgow.ac.uk>

Repository CRAN

Date/Publication 2023-08-27 22:40:06 UTC

R topics documented:

`dgin` *Density for the generalized inverse normal distribution*

Description

Density for the generalized inverse normal distribution

Usage

```
dgin(z, alpha, mu, tau, log = TRUE, quasi = FALSE)
```

Arguments

`z`	quantile.
`alpha`	degrees-of-freedom parameter.
`mu`	similar to location parameter, controls asymmetry of the distribution.
`tau`	similar to scale parameter, controls spread of the distribution.
`log`	logical; should the log of the density be returned? Defaults to TRUE.
`quasi`	logical; should the quasi-density value be returned? Defaults to FALSE.

Details

Currently, only scalars are supported for the quantile and parameter values. Density is supported on the entire real line, `z` and `mu` can take any value in $(-\infty, \infty)$. Density is only defined for parameter values `alpha` > 1 or `tau` > 0, so it is set to 0 outside of these values. The quasi-density or kernel is the density without the normalization constant, use `quasi = TRUE` for this behavior.

Value

Numeric scalar with density.

Examples

```
# Computing (log) density
dgin(z = 1, alpha = 3, mu = 1, tau = 1, log = TRUE, quasi = FALSE)

# Generalized inverse normal density with alpha = 5, mu = 0, tau = 1
z_vals <- seq(-5, 5, length.out = 200)
fz_unc <- sapply(z_vals, function(z) dgin(z, 5, 0, 1, FALSE))
plot(z_vals, fz_unc, type = "l", xlab = 'Values', ylab = 'Density')
```

dtgin — *Density for the generalized inverse normal distribution truncated to the positive or negative reals*

Description

Density for the generalized inverse normal distribution truncated to the positive or negative reals

Usage

```
dtgin(
  z,
  alpha,
  mu,
  tau,
  sign = TRUE,
  log = TRUE,
  quasi = FALSE,
  method = "Fortran"
)
```

Arguments

`z`	quantile.
`alpha`	degrees-of-freedom parameter.
`mu`	similar to location parameter, controls asymmetry of the distribution.
`tau`	similar to scale parameter, controls spread of the distribution.
`sign`	logical. `TRUE` implies truncation to positive numbers (`z` > 0) and `FALSE` to negative numbers (`z` < 0).
`log`	logical; should the log of the density be returned? Defaults to TRUE.
`quasi`	logical; should the quasi-density value be returned? Defaults to FALSE.
`method`	string with the method used to compute the parabolic cylinder function in the normalization constant. `method = "Fortran"` uses a compiled Fortran version, which is the default. `method = "R"` uses an R translation of this function.

Details

Currently, only scalars are supported for the quantile and parameter values. Density is supported on the positive reals (`z` > 0) when `sign = TRUE` and to negative reals (`z` < 0) when `sign = FALSE`. `mu` can take any value in $(-\infty, \infty)$. Density is only defined for parameter values `alpha` > 1 or `tau` > 0, so it is set to 0 outside of these values. The quasi-density or kernel is the density without the normalization constant, use `quasi = TRUE` for this behavior.

Value

Numeric scalar with density.

Examples

```
# Computing (log) truncated densities
dtgin(z = 1, alpha = 3, mu = 1, tau = 1, sign = TRUE, log = TRUE, quasi = FALSE)
dtgin(z = -1, alpha = 3, mu = -1, tau = 1, sign = FALSE, log = TRUE, quasi = FALSE)

# Generalized inverse normal density with alpha = 5, mu = 0, tau = 1
n_values <- 200
z_vals <- seq(-5, 5, length.out = n_values)

# Truncated to positive reals (z > 0)
fz_p <- sapply(z_vals[z_vals > 0], function(z) dtgin(z, 5, 0, 1, TRUE, FALSE))
fz_p <- c(rep(0, n_values - sum(z_vals > 0)), fz_p)
plot(z_vals, fz_p, type = "l", xlab = 'Values', ylab = 'Density')

# Truncated to positive reals (z < 0)
fz_n <- sapply(z_vals[z_vals < 0], function(z) dtgin(z, 5, 0, 1, FALSE, FALSE))
fz_n <- c(fz_n, rep(0, n_values - sum(z_vals < 0)))
plot(z_vals, fz_n, type = "l", xlab = 'Values', ylab = 'Density')

# Both truncated densities together
plot(z_vals, fz_p, type = "l", xlab = 'Values', ylab = 'Density')
lines(z_vals, fz_n, col = 'blue', lty = 2)
legend('topright', legend = c('z > 0', 'z < 0'),
       col = c('black', 'blue'), lty = 1:2)
```

---

rgin — *Generating random numbers from the generalized inverse normal distribution*

Description

Generating random numbers from the generalized inverse normal distribution

Usage

```
rgin(size, alpha, mu, tau, algo = "hormann", method = "Fortran")
```

Arguments

`size`	number of desired draws. Output is numpy vector of length equal to size.
`alpha`	degrees-of-freedom parameter.
`mu`	similar to location parameter, controls asymmetry of the distribution.
`tau`	similar to scale parameter, controls spread of the distribution.
`algo`	string with desired algorithm to compute minimal bounding rectangle. If "hormann", use the method from Hörmann and Leydold (2014). When "leydold", use the one from Leydold (2001). Defaults to "hormann" and returns an error for any other values.

`method`	string with the method used to compute the parabolic cylinder function in the normalization constant. `method = "Fortran"` uses a compiled Fortran version, which is the default. `method = "R"` uses an R translation of this function.

Details

Currently, only values of `alpha` > 2 are supported. For Bayesian posterior sampling, `alpha` is always larger than 2 even for non-informative priors. The algorithm requires calculating the probability of truncation region (either `z` < 0 or `z` > 0). It is more stable to compute a probability bounded away from 0. As `mu` controls asymmetry, when `mu` > 0, P(truncation region) = P(z > 0) >= 50%, and this probability is computed. If `mu` < 0, P(z < 0) >= 50% and this region's probability is used.

Value

Numeric vector of length `size`.

Examples

```
# Generate 1000 values from the distribution with alpha = 5, mu = 0, tau = 1
set.seed(123456)
z_unc <- rgin(1000, 5, 0, 1)

# Compare histogram to true density
z_vals <- seq(-5, 5, length.out = 200)
fz_unc <- sapply(z_vals, function(z) dgin(z, 5, 0, 1, FALSE))
temp <- hist(z_unc, breaks = 200, plot = FALSE)
plot(temp, freq = FALSE, xlim = c(-5, 5), ylim = range(c(fz_unc, temp$density)),
     main = '', xlab = 'Values', ylab = 'Density', col = 'blue')
lines(z_vals, fz_unc, col = 'red', lwd = 2)
```

`rtgin` *Generating random numbers from the generalized inverse normal distribution truncated to the positive or negative reals*

Description

Generating random numbers from the generalized inverse normal distribution truncated to the positive or negative reals

Usage

```
rtgin(
  size,
  alpha,
  mu,
  tau,
  sign,
  algo = "hormann",
  method = "Fortran",
  verbose = FALSE
)
```

Arguments

`size`	number of desired draws. Output is numpy vector of length equal to size.
`alpha`	degrees-of-freedom parameter.
`mu`	similar to location parameter, controls asymmetry of the distribution.
`tau`	similar to scale parameter, controls spread of the distribution.
`sign`	logical. `TRUE` implies truncation to positive numbers (z > 0) and `FALSE` to negative numbers (z < 0).
`algo`	string with desired algorithm to compute minimal bounding rectangle. If "hormann", use the method from Hörmann and Leydold (2014). When "leydold", use the one from Leydold (2001). Defaults to "hormann" and returns an error for any other values.
`method`	string with the method used to compute the parabolic cylinder function in the normalization constant. `method = "Fortran"` uses a compiled Fortran version, which is the default. `method = "R"` uses an R translation of this function.
`verbose`	logical; should the acceptance rate from the ratio-of-uniforms method be provided along with additional information? Defaults to FALSE.

Details

Currently, only values of `alpha` > 2 are supported. For Bayesian posterior sampling, `alpha` is always larger than 2 even for non-informative priors. Generate from positive region (z > 0) hen `sign = TRUE`, and from negative region (z < 0) when `sign = FALSE`. When `verbose = TRUE`, a list is returned containing the actual draw in `value`, as well as average acceptance rate `avg_arate` and total number of acceptance-rejection steps `ARiters`.

Value

If `verbose = FALSE` (default), a numeric vector of length `size`. Otherwise, a list with components `value`, `avg_arate`, and `ARiters`

Examples

```
# Generate 1000 values from the truncated distributions with alpha = 5, mu = 0, tau = 1
set.seed(123456)
n_draws <- 1000
z_p <- rtgin(n_draws, 5, 0, 1, TRUE)
z_n <- rtgin(n_draws, 5, 0, 1, FALSE)

# Compare generation from truncation to positive reals with true density
n_values <- 200
z_vals <- seq(-5, 5, length.out = n_values)
fz_p <- sapply(z_vals[z_vals > 0], function(z) dtgin(z, 5, 0, 1, TRUE, FALSE))
fz_p <- c(rep(0, n_values - sum(z_vals > 0)), fz_p)
```

```
temp <- hist(z_p, breaks = 100, plot = FALSE)
plot(temp, freq = FALSE, xlim = c(-5, 5), ylim = range(c(fz_p, temp$density)),
     main = '', xlab = 'Values', ylab = 'Density', col = 'blue')
lines(z_vals, fz_p, col = 'red', lwd = 2)

# Compare generation from truncation to negative reals with true density
fz_n <- sapply(z_vals[z_vals < 0], function(z) dtgin(z, 5, 0, 1, FALSE, FALSE))
fz_n <- c(fz_n, rep(0, n_values - sum(z_vals < 0)))
temp <- hist(z_n, breaks = 100, plot = FALSE)
plot(temp, freq = FALSE, xlim = c(-5, 5), ylim = range(c(fz_n, temp$density)),
     main = '', xlab = 'Values', ylab = 'Density', col = 'blue')
lines(z_vals, fz_n, col = 'red', lwd = 2)

# verbose = TRUE provides info on the acceptance rate of the
# ratio-of-uniforms acceptance-rejection method for sampling the variables
draw_list <- rtgin(50, 5, 0, 1, sign = TRUE, verbose = TRUE)
draw_list$ARiters      # Acceptance-Rejection iterations
draw_list$avg_arate    # Average of 1/ARiters
```

Index