

# Package ‘tsdistributions’

April 17, 2025

**Type** Package

**Title** Location Scale Standardized Distributions

**Version** 1.0.3

**Maintainer** Alexios Galanos <alexios@4dscape.com>

**Depends** R (>= 3.5.0), methods, tsmethods

**LinkingTo** Rcpp, TMB, RcppEigen

**Imports** Rcpp, TMB (>= 1.7.20), Rdpack, GeneralizedHyperbolic, KernSmooth, SkewHyperbolic, mev, stats, utils, data.table, Rsolnp, sandwich, future.apply, future, progressr

**Description** Location-Scale based distributions parameterized in terms of mean, standard deviation, skew and shape parameters and estimation using automatic differentiation. Distributions include the Normal, Student and GED as well as their skewed variants ('Fernandez and Steel'), the 'Johnson SU', and the Generalized Hyperbolic. Also included is the semi-parametric piece wise distribution ('spd') with Pareto tails and kernel interior.

**License** GPL-2

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**VignetteBuilder** knitr

**RdMacros** Rdpack

**URL** <https://www.nopredict.com/packages/tsdistributions>,  
<https://github.com/tsmodels/tsdistributions>

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**NeedsCompilation** yes

**Author** Alexios Galanos [aut, cre, cph]  
(<<https://orcid.org/0009-0000-9308-0457>>)

**Repository** CRAN

**Date/Publication** 2025-04-17 09:50:02 UTC

## Contents

AIC.tsdistribution.estimate	2
authorized_domain	3
BIC.tsdistribution.estimate	4
bread.tsdistribution.spdestimate	4
coef.tsdistribution.estimate	5
ddist	5
dged	7
dgh	8
dghst	9
dghyp	10
distribution_bounds	11
distribution_modelspec	12
djsu	13
dnig	14
dsged	15
dskewness	16
dsnrm	16
dspd	17
dsstd	18
dstd	19
estfun.tsdistribution.estimate	20
estimate.tsdistribution.spdspec	20
estimate.tsdistribution.spec	21
logLik.tsdistribution.estimate	22
nigtransform	23
print.summary.tsdistribution	23
print.summary.tsdistribution.profile	24
spd_modelspec	25
summary.tsdistribution.estimate	25
summary.tsdistribution.profile	26
summary.tsdistribution.spdestimate	27
tsmoments.tsdistribution.estimate	27
tsprofile.tsdistribution.spec	28
vcov.tsdistribution.estimate	29
<b>Index</b>	<b>30</b>

---

AIC.tsdistribution.estimate

*Akaike's An Information Criterion*

---

## Description

Extract the AIC from an estimated model.

**Usage**

```
## S3 method for class 'tsdistribution.estimate'
AIC(object, ..., k = 2)

## S3 method for class 'tsdistribution.spdestimate'
AIC(object, ..., k = 2)
```

**Arguments**

object            an object of class “tsdistribution.estimate”.

...                not currently used.

k                  the penalty per parameter to be used; the default  $k = 2$  is the classical AIC.

**Value**

The AIC value (scalar).

---

authorized_domain	<i>Distribution Authorized Domain</i>
-------------------	---------------------------------------

---

**Description**

Calculated the region of Skewness-Kurtosis for which a density exists.

**Usage**

```
authorized_domain(distribution, max_kurt = 30, n = 25, lambda = 1)
```

**Arguments**

distribution    a valid distribution with skew and shape parameters.

max\_kurt        the maximum kurtosis for which to determine the bounds for the skewness-kurtosis domain.

n                the number of points between the lower and upper bounds of the skew and shape parameters for which to evaluate the skewness and excess kurtosis. This determines the kurtosis interval ( $3 - \text{max\_kurt}$ ) for which to calculate (solver based) the maximum skewness.

lambda          additional shape parameter for the Generalized Hyperbolic distribution.

**Value**

A list with the lower half of the skewness and kurtosis values.

BIC.tsdistribution.estimate

*Bayesian Information Criterion*

---

### Description

Extract the BIC from an estimated model.

### Usage

```
## S3 method for class 'tsdistribution.estimate'  
BIC(object, ...)
```

```
## S3 method for class 'tsdistribution.spdestimate'  
BIC(object, ...)
```

### Arguments

object            an object of class “tsdistribution.estimate”.  
...                not currently used.

### Value

The BIC value (scalar).

---

bread.tsdistribution.spdestimate

*Bread Method*

---

### Description

Bread Method

### Usage

```
## S3 method for class 'tsdistribution.spdestimate'  
bread(x, ...)
```

```
## S3 method for class 'tsdistribution.estimate'  
bread(x, ...)
```

### Arguments

x                 an object of class “tsdistribution.estimate”.  
...                not currently used.

**Value**

The analytic hessian of the model.

**Author(s)**

Alexios Galanos

---

coef.tsdistribution.estimate  
*Extract Model Coefficients*

---

**Description**

Extract Model Coefficients

**Usage**

```
## S3 method for class 'tsdistribution.estimate'  
coef(object, ...)  
  
## S3 method for class 'tsdistribution.spdestimate'  
coef(object, ...)
```

**Arguments**

object            an object of class tsdistribution.estimate.  
...                other arguments.

**Value**

A vector of the estimated model coefficients.

---

ddist             *Distributions pqdr wrapper*

---

**Description**

Density, distribution, quantile function and random number generation for all the distributions in the package.

**Usage**

```
ddist(  
  distribution = "norm",  
  x,  
  mu = 0,  
  sigma = 1,  
  skew = 1,  
  shape = 5,  
  lambda = -0.5,  
  log = FALSE  
)
```

```
pdist(  
  distribution = "norm",  
  q,  
  mu = 0,  
  sigma = 1,  
  skew = 1,  
  shape = 5,  
  lambda = -0.5,  
  lower_tail = TRUE,  
  log = FALSE  
)
```

```
qdist(  
  distribution = "norm",  
  p,  
  mu = 0,  
  sigma = 1,  
  skew = 1,  
  shape = 5,  
  lambda = -0.5,  
  lower_tail = TRUE,  
  log = FALSE  
)
```

```
rdist(  
  distribution = "norm",  
  n,  
  mu = 0,  
  sigma = 1,  
  skew = 1,  
  shape = 5,  
  lambda = -0.5  
)
```

**Arguments**

distribution	a valid distribution.
x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.
shape	shape parameter.
lambda	additional shape parameter for the Generalized Hyperbolic distribution.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dged	<i>Generalized Error Distribution</i>
------	---------------------------------------

---

**Description**

Density, distribution, quantile function and random number generation for the generalized error distribution parameterized in terms of mean, standard deviation and shape parameters.

**Usage**

```
dged(x, mu = 0, sigma = 1, shape = 2, log = FALSE)
pged(q, mu = 0, sigma = 1, shape = 2, lower_tail = TRUE, log = FALSE)
qged(p, mu = 0, sigma = 1, shape = 2, lower_tail = TRUE, log = FALSE)
rged(n, mu = 0, sigma = 1, shape = 2)
```

**Arguments**

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
shape	shape parameter.
log	(logical) if TRUE, probabilities p are given as log(p).





**Arguments**

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.
shape	shape parameter.
lambda	additional shape parameter determining subfamilies of this distributions.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dghst

*Generalized Hyperbolic Skewed Student Distribution*


---

**Description**

Density, distribution, quantile function and random number generation for the generalized hyperbolic skew student distribution parameterized in terms of mean, standard deviation, skew and shape parameters.

**Usage**

```
dghst(x, mu = 0, sigma = 1, skew = 1, shape = 8, log = FALSE)
```

```
rgkst(n, mu = 0, sigma = 1, skew = 1, shape = 8)
```

```
pghst(
  q,
  mu = 0,
  sigma = 1,
  skew = 1,
  shape = 8,
  lower_tail = TRUE,
  log = FALSE
)
```

```
qghst(
  p,
```

```

mu = 0,
sigma = 1,
skew = 1,
shape = 8,
lower_tail = TRUE,
log = FALSE
)

```

### Arguments

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.
shape	shape parameter.
log	(logical) if TRUE, probabilities p are given as log(p).
n	Number of observations.
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.

### Value

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dghyp	<i>Generalized Hyperbolic Distribution (alpha-beta-delta-mu parameterization)</i>
-------	---

---

### Description

Density, distribution, quantile function and random number generation for the generalized hyperbolic distribution using the alpha-beta-delta-mu-lambda parameterization.

### Usage

```
dghyp(x, alpha = 1, beta = 0, delta = 1, mu = 0, lambda = 1, log = FALSE)
```

```

pghyp(
  q,
  alpha = 1,
  beta = 0,
  delta = 1,
  mu = 0,
  lambda = 1,
)

```

```

    lower_tail = TRUE,
    log = FALSE
  )

  qghyp(
    p,
    alpha = 1,
    beta = 0,
    delta = 1,
    mu = 0,
    lambda = 1,
    lower_tail = TRUE,
    log = FALSE
  )

  rghyp(n, alpha = 1, beta = 0, delta = 1, mu = 0, lambda = 1)

```

### Arguments

x, q	vector of quantiles.
alpha	tail parameter.
beta	skewness parameter.
delta	scale parameter.
mu	location parameter.
lambda	additional shape parameter determining subfamilies of this distributions.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

### Value

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

distribution\_bounds     *Distribution Bounds*

---

### Description

Distribution Bounds

### Usage

```
distribution_bounds(distribution = "norm")
```

**Arguments**

distribution    A valid distribution

**Details**

Returns the upper a lower bounds for the parameters of a distribution.

**Value**

A data.table of the parameters and their default bounds.

---

distribution\_modelspec

*Specification of distribution model*

---

**Description**

Specification of distribution model

**Usage**

```
distribution_modelspec(y, distribution = "norm", ...)
```

**Arguments**

y                    a numeric vector

distribution    the type of distribution. Valid choices are norm (Normal), snorm (Skew Normal), std (Student), sstd (Skew Student), ged (Generalized Error), sged (Skew Generalized Error), nig (Normal Inverse Gaussian), gh (Generalized Hyperbolic), ghst (Generalized Hyperbolic Skew Student) and jsu (Johnson's SU).

...                not currently used

**Details**

All distributions are parameterized in terms of their mean ('mu'), standard deviation 'sigma', skew 'skew' and shape 'shape' parameters. Additionally, for the Generalized Hyperbolic distribution, there is an extra shape parameter "lambda" arising from the GIG mixing distribution. Parameters can be fixed post initialization by setting setting specific values to the 'value' column in the parmatrix table and setting the 'estimate' variable to 0 (instead of 1).

**Value**

An object of class "tsdistribution.spec".

## Examples

```
spec <- distribution_modelspec(rnorm(1000), distribution = "gh")
# fix lambda and shape
spec$parmatrix[parameter == 'lambda', value := 30]
spec$parmatrix[parameter == 'lambda', estimate := 0]
```

---

djsu

*Johnson's SU Distribution*

---

## Description

Density, distribution, quantile function and random number generation for Johnson's SU distribution parameterized in terms of mean, standard deviation, skew and shape parameters.

## Usage

```
djsu(x, mu = 0, sigma = 1, skew = 1, shape = 0.5, log = FALSE)
```

```
pjsu(
  q,
  mu = 0,
  sigma = 1,
  skew = 1,
  shape = 0.5,
  lower_tail = TRUE,
  log = FALSE
)
```

```
qjsu(
  p,
  mu = 0,
  sigma = 1,
  skew = 1,
  shape = 0.5,
  lower_tail = TRUE,
  log = FALSE
)
```

```
rjsu(n, mu = 0, sigma = 1, skew = 1, shape = 0.5)
```

## Arguments

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.

shape	shape parameter.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dnig

*Normal Inverse Gaussian Distribution*


---

**Description**

Density, distribution, quantile function and random number generation for the normal inverse gaussian distribution generalized parameterized in terms of mean, standard deviation, skew and shape parameters.

**Usage**

```
dnig(x, mu = 0, sigma = 1, skew = 0, shape = 1, log = FALSE)
```

```
pnig(q, mu = 0, sigma = 1, skew = 0, shape = 1, lower_tail = TRUE, log = FALSE)
```

```
qnig(p, mu = 0, sigma = 1, skew = 0, shape = 1, lower_tail = TRUE, log = FALSE)
```

```
rnig(n, mu = 0, sigma = 1, skew = 0, shape = 1)
```

**Arguments**

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.
shape	shape parameter.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dsged

*Skewed Generalized Error Distribution of Fernandez and Steel*


---

**Description**

Density, distribution, quantile function and random number generation for the skewed generalized error distribution parameterized in terms of mean, standard deviation, skew and shape parameters.

**Usage**

```
dsged(x, mu = 0, sigma = 1, skew = 1.5, shape = 2, log = FALSE)
```

```
psged(
  q,
  mu = 0,
  sigma = 1,
  skew = 1.5,
  shape = 2,
  lower_tail = TRUE,
  log = FALSE
)
```

```
qsged(
  p,
  mu = 0,
  sigma = 1,
  skew = 1.5,
  shape = 2,
  lower_tail = TRUE,
  log = FALSE
)
```

```
rsged(n, mu = 0, sigma = 1, skew = 1.5, shape = 2)
```

**Arguments**

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.
shape	shape parameter.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dskewness	<i>Distribution skewness and kurtosis</i>
-----------	---

---

**Description**

Calculates the skewness and excess kurtosis of the distribution given a set of parameters.

**Usage**

```
dskewness(distribution = "norm", skew = 1, shape = 5, lambda = -0.5)
```

```
dkurtosis(distribution = "norm", skew = 1, shape = 5, lambda = -0.5)
```

**Arguments**

distribution	a valid distribution.
skew	skew parameter.
shape	shape parameter.
lambda	additional shape parameter for the Generalized Hyperbolic distribution.

**Value**

A numeric value for the skewness and excess kurtosis.

---

dsnorm	<i>Skewed Normal Distribution of Fernandez and Steel</i>
--------	--

---

**Description**

Density, distribution, quantile function and random number generation for the skewed normal distribution parameterized in terms of mean, standard deviation and skew parameters.

**Usage**

```
dsnorm(x, mu = 0, sigma = 1, skew = 1.5, log = FALSE)
```

```
psnorm(q, mu = 0, sigma = 1, skew = 1.5, lower_tail = TRUE, log = FALSE)
```

```
qsnorm(p, mu = 0, sigma = 1, skew = 1.5, lower_tail = TRUE, log = FALSE)
```

```
rsnorm(n, mu = 0, sigma = 1, skew = 1.5)
```



**Arguments**

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	Number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dspd	<i>Semi-Parametric Distribution</i>
------	-------------------------------------

---

**Description**

Density, distribution, quantile function and random number generation for the semi parametric distribution (spd) which has generalized Pareto tails and kernel fitted interior.

**Usage**

```
dspd(x, object, linear = TRUE, log = FALSE)
pspd(q, object, linear = TRUE, lower_tail = TRUE)
qspd(p, object, linear = TRUE, lower_tail = TRUE)
rspd(n, object, linear = TRUE)
```

**Arguments**

x, q	vector of quantiles.
object	an object of class "tsdistribution.spdestimate" returned from calling <a href="#">estimate.tsdistribution.spdspe</a>
linear	logical, if TRUE (default) interior smoothing function uses linear interpolation rather than constant.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	Number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

 dsstd
 

---



---

*Skewed Student Distribution of Fernandez and Steel*


---

**Description**

Density, distribution, quantile function and random number generation for the skewed student distribution parameterized in terms of mean, standard deviation, skew and shape parameters.

**Usage**

```
dsstd(x, mu = 0, sigma = 1, skew = 1.5, shape = 5, log = FALSE)
```

```
psstd(
  q,
  mu = 0,
  sigma = 1,
  skew = 1.5,
  shape = 5,
  lower_tail = TRUE,
  log = FALSE
)
```

```
qsstd(
  p,
  mu = 0,
  sigma = 1,
  skew = 1.5,
  shape = 5,
  lower_tail = TRUE,
  log = FALSE
)
```

```
rsstd(n, mu = 0, sigma = 1, skew = 1.5, shape = 5)
```

**Arguments**

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
skew	skew parameter.
shape	shape parameter.

log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

---

dstd *Student Distribution*

---

**Description**

Density, distribution, quantile function and random number generation for the student distribution parameterized in terms of mean, standard deviation and shape parameters.

**Usage**

```
dstd(x, mu = 0, sigma = 1, shape = 5, log = FALSE)
pstd(q, mu = 0, sigma = 1, shape = 5, lower_tail = TRUE, log = FALSE)
qstd(p, mu = 0, sigma = 1, shape = 5, lower_tail = TRUE, log = FALSE)
rstd(n, mu = 0, sigma = 1, shape = 5)
```

**Arguments**

x, q	vector of quantiles.
mu	mean.
sigma	standard deviation.
shape	shape parameter.
log	(logical) if TRUE, probabilities p are given as log(p).
lower_tail	if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

**Value**

d gives the density, p gives the distribution function, q gives the quantile function and r generates random deviates. Output depends on x or q length, or n for the random number generator.

estfun.tsdistribution.estimate  
*Score Method*

---

**Description**

Score Method

**Usage**

```
## S3 method for class 'tsdistribution.estimate'  
estfun(x, ...)
```

**Arguments**

x                    an object of class "tsdistribution.estimate".  
...                   not currently used.

**Details**

The function returns the scores of likelihood at the optimal solution.

**Value**

The score matrix.

**Author(s)**

Alexios Galanos

---

estimate.tsdistribution.spdspec  
*Estimates the parameters of a semi-parametric distribution.*

---

**Description**

Estimates the parameters of a semi-parametric distribution.

**Usage**

```
## S3 method for class 'tsdistribution.spdspec'  
estimate(object, method = "pwm", ...)
```

**Arguments**

object	an object of class “tsdistribution.spdspec”.
method	a choice of “Grimshaw”, “obre” or “nlm” from <code>fit.gpd</code> or “pwm” for the probability weighted moments estimator.
...	additional parameters passed to the gpd estimation function.

**Details**

The estimation defaults to the Probability Weighted Moments (pwm) of Hosking (1985), and alternative methods are provided via the “mev” package. For the interior of the distribution, the `bkde` function is used to calculate the kernel density.

**Value**

An object of class “tsdistribution.spdestimate” with slots for the upper, lower and interior kernel fitted values.

**References**

Hosking JRM, Wallis JR, Wood EF (1985). “Estimation of the generalized extreme-value distribution by the method of probability-weighted moments.” *Technometrics*, **27**(3), 251–261.

---

```
estimate.tsdistribution.spec
```

*Estimates the parameters of a distribution using autodiff.*

---

**Description**

Estimates the parameters of a distribution using autodiff.

**Usage**

```
## S3 method for class 'tsdistribution.spec'
estimate(
  object,
  solver = "nlminb",
  control = list(trace = 0, eval.max = 300, iter.max = 500),
  use_hessian = TRUE,
  ...
)
```

**Arguments**

object	an object of class “tsdistribution.spec”.
solver	only “nlminb” currently supported.
control	solver control parameters.
use_hessian	whether to use the hessian in the calculation.
...	additional parameters passed to the estimation function

**Details**

The estimation makes use of the TMB package for minimizing the negative of the log-likelihood using automatic differentiation.

**Value**

An object of class “tsdistribution.estimate” with slots for the estimated coefficients, gradients, scores etc.

---

logLik.tsdistribution.estimate  
*Extract Log-Likelihood*

---

**Description**

Extract Log-Likelihood

**Usage**

```
## S3 method for class 'tsdistribution.estimate'  
logLik(object, ...)  
  
## S3 method for class 'tsdistribution.spdestimate'  
logLik(object, ...)
```

**Arguments**

object	an object of class tsdistribution.estimate.
...	other arguments.

**Value**

An object of class logLik. This is a number with at least one attribute, “df” (degrees of freedom), giving the number of (estimated) parameters in the model.

---

nigtransform	<i>Parameter Transformation</i>
--------------	---------------------------------

---

**Description**

Transforms parameters from standardized representation to distribution specific representation for the nig and gh distributions.

**Usage**

```
nigtransform(mu = 0, sigma = 1, skew = 0, shape = 3)
```

```
ghyptransform(mu = 0, sigma = 1, skew = 0, shape = 3, lambda = -0.5)
```

**Arguments**

mu	mean.
sigma	standard deviation.
skew	skew parameter.
shape	shape parameter.
lambda	additional shape parameter for the Generalized Hyperbolic distribution.

**Value**

The (alpha, beta, delta, mu) representation.

---

```
print.summary.tsdistribution
```

*Model Estimation Summary Print method*

---

**Description**

Print method for class "summary.tsdistribution"

**Usage**

```
## S3 method for class 'summary.tsdistribution'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"),
  table.caption = paste0(toupper(x$distribution), " Model Summary\n"),
  ...
)
```

```
## S3 method for class 'summary.spd'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"),
  table.caption = paste0(toupper(x$distribution), " Model Summary\n"),
  ...
)
```

### Arguments

x	an object of class “summary.tsdistribution”.
digits	integer, used for number formatting. Optionally, to avoid scientific notation, set ‘options(scipen=999)’.
signif.stars	logical. If TRUE, ‘significance stars’ are printed for each coefficient.
table.caption	an optional string for the table caption.
...	not currently used.

### Value

Console output of the object summary.

---

```
print.summary.tsdistribution.profile
      Profile Summary Print method
```

---

### Description

Print method for class “summary.tsdistribution.profile”

### Usage

```
## S3 method for class 'summary.tsdistribution.profile'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

### Arguments

x	an object of class “summary.tsdistribution.profile”.
digits	integer, used for number formatting. Optionally, to avoid scientific notation, set ‘options(scipen=999)’.
...	not currently used.

### Value

Invisibly returns the original summary object and prints out to the console.



---

spd\_modelspec                      *Specification of a semi-parametric distribution model*

---

**Description**

Specification of a semi-parametric distribution model

**Usage**

```
spd_modelspec(
  y,
  lower = 0.1,
  upper = 0.9,
  kernel_type = c("normal", "box", "epanech", "biweight", "triweight"),
  ...
)
```

**Arguments**

y	a numeric vector
lower	the probability for the lower GPD tail.
upper	the probability for the upper GPD tail.
kernel_type	the choice of the kernel to use from the <a href="#">bkde</a> function.
...	not currently used

**Value**

An object of class “tsdistribution.spd\_spec”.

**Examples**

```
spec <- spd_modelspec(rnorm(1000))
```

---

summary.tsdistribution.estimate  
*Summary of estimated distribution*

---

**Description**

Summary of estimated distribution

**Usage**

```
## S3 method for class 'tsdistribution.estimate'
summary(object, digits = 4, vcov_type = "H", ...)
```

**Arguments**

object	an object of class <code>tsdistribution.estimate</code> .
digits	the number of significant digits to use when printing,.
vcov_type	the type of standard errors based on the <code>vcov</code> estimate (see <code>vcov</code> ).
...	additional parameters passed to the summary method.

**Value**

A list of summary statistics of the fitted model given in object.

---

```
summary.tsdistribution.profile
```

*Distribution Profile Summary*

---

**Description**

Summary method for class “`tsdistribution.profile`”

**Usage**

```
## S3 method for class 'tsdistribution.profile'
summary(object, digits = 4, measure = "RMSE", ...)
```

**Arguments**

object	an object of class “ <code>tsdistribution.profile</code> ”.
digits	integer, used for number formatting. Optionally, to avoid scientific notation, set ‘ <code>options(scipen=999)</code> ’.
measure	either one of the 3 included measure in the summary slot of the returned object “ <code>RMSE</code> ”, “ <code>MAE</code> ” or “ <code>MAPE</code> ”, else any other user calculated measure which has been generated in the summary table post processing.
...	not currently used.

**Value**

A list with summary information of class “`summary.tsdistribution.profile`”, including a table with each actual parameter against the measure chosen across each size in the profile.

---

```
summary.tsdistribution.spdestimate
      Summary of estimated SPD distribution
```

---

**Description**

Summary of estimated SPD distribution

**Usage**

```
## S3 method for class 'tsdistribution.spdestimate'
summary(object, ...)
```

**Arguments**

object            an object of class “tsdistribution.spdestimate”.  
 ...              additional parameters passed to the summary method.

**Details**

The standard errors assume a block diagonal covariance structure between the upper and lower Generalized Pareto Tails.

**Value**

A list of summary statistics of the fitted model given in object.

---

```
tsmoments.tsdistribution.estimate
      Extract the moments of an estimated distribution
```

---

**Description**

Extract the moments of an estimated distribution

**Usage**

```
## S3 method for class 'tsdistribution.estimate'
tsmoments(object, ...)
```

**Arguments**

object            an object of class tsdistribution.estimate.  
 ...              other arguments.

**Value**

A vector of the first four moments of the distribution based on the estimated parameters. The kurtosis represents the value in excess of 3.

---

 tsprofile.tsdistribution.spec

*Model Parameter Profiling*

---

**Description**

Profiles the model parameters under the specified distribution.

**Usage**

```
## S3 method for class 'tsdistribution.spec'
tsprofile(
  object,
  nsim = 100,
  sizes = c(800, 1000, 1500, 2000, 3000),
  seed = NULL,
  trace = FALSE,
  ...
)
```

**Arguments**

object	an object of class “tsdistribution.spec” with pre-set parameters.
nsim	the number of paths to generate.
sizes	a vector of data sizes for which to simulate and estimate.
seed	an object specifying if and how the random number generator should be initialized. See the simulate documentation for more details.
trace	whether to show the progress bar. The user is expected to have set up appropriate handlers for this using the “progressr” package.
...	not currently used.

**Details**

The function profiles the parameters of a model by simulating and then estimating multiple paths from the assumed distribution. This makes it possible to obtain a better understanding of the convergence properties (RMSE) of each parameter under different data sizes.

**Value**

An object of class “tsdistribution.profile”.

**Note**

The function can use parallel functionality as long as the user has set up a [plan](#) using the future package.

---

vcov.tsdistribution.estimate

*The Covariance Matrix of the Estimated Parameters*

---

**Description**

The Covariance Matrix of the Estimated Parameters

**Usage**

```
## S3 method for class 'tsdistribution.estimate'
vcov(object, adjust = FALSE, type = c("H", "OP", "QMLE", "NW"), ...)

## S3 method for class 'tsdistribution.spdestimate'
vcov(object, ...)
```

**Arguments**

object	an object of class <code>tsdistribution.estimate</code>
adjust	logical. Should a finite sample adjustment be made? This amounts to multiplication with $n/(n-k)$ where $n$ is the number of observations and $k$ the number of estimated parameters.
type	valid choices are “H” for using the analytic hessian for the ‘bread’, “OP” for the outer product of gradients, “QMLE” for the Quasi-ML sandwich estimator (Huber-White), and “NW” for the Newey-West adjusted sandwich estimator (a HAC estimator).
...	additional parameters passed to the Newey-West bandwidth function to determine the optimal lags.

**Value**

The variance-covariance matrix of the estimated parameters.

# Index

AIC (AIC.tsdistribution.estimate), 2  
AIC.tsdistribution.estimate, 2  
authorized\_domain, 3

BIC (BIC.tsdistribution.estimate), 4  
BIC.tsdistribution.estimate, 4  
bkde, 21, 25  
bread  
    (bread.tsdistribution.spdestimate),  
    4  
bread.tsdistribution.spdestimate, 4

coef (coef.tsdistribution.estimate), 5  
coef.tsdistribution.estimate, 5

ddist, 5  
dged, 7  
dgh, 8  
dghst, 9  
dghyp, 10  
distribution\_bounds, 11  
distribution\_modelspec, 12  
djsu, 13  
dkurtosis (dskewness), 16  
dnig, 14  
dsged, 15  
dskewness, 16  
dsnrm, 16  
dspd, 17  
dsstd, 18  
dstd, 19

estfun  
    (estfun.tsdistribution.estimate),  
    20  
estfun.tsdistribution.estimate, 20  
estimate  
    (estimate.tsdistribution.spec),  
    21  
estimate.tsdistribution.spdspec, 17, 20

estimate.tsdistribution.spec, 21  
fit.gpd, 21

ghyptransform (nigtransform), 23

logLik  
    (logLik.tsdistribution.estimate),  
    22  
logLik.tsdistribution.estimate, 22

nigtransform, 23

pdist (ddist), 5  
pged (dged), 7  
pgh (dgh), 8  
pghst (dghst), 9  
pghyp (dghyp), 10  
pjsu (djsu), 13  
plan, 29  
pnig (dnig), 14  
print.summary.spd  
    (print.summary.tsdistribution),  
    23  
print.summary.tsdistribution, 23  
print.summary.tsdistribution.profile,  
    24  
psged (dsged), 15  
psnorm (dsnrm), 16  
pspd (dspd), 17  
psstd (dsstd), 18  
pstd (dstd), 19

qdist (ddist), 5  
qged (dged), 7  
qgh (dgh), 8  
qghst (dghst), 9  
qghyp (dghyp), 10  
qjsu (djsu), 13  
qnig (dnig), 14  
qsged (dsged), 15

qsnorm (dsnrm), 16  
qspd (dspd), 17  
qsstd (dsstd), 18  
qstd (dstd), 19

rdist (ddist), 5  
rged (dged), 7  
rgh (dgh), 8  
rghst (dghst), 9  
rghyp (dghyp), 10  
rjsu (djsu), 13  
rnig (dnig), 14  
rsged (dsGED), 15  
rsnorm (dsnrm), 16  
rspd (dspd), 17  
rsstd (dsstd), 18  
rstd (dstd), 19

spd\_modelspec, 25  
summary  
    (summary.tsdistribution.estimate),  
    25  
summary.tsdistribution.estimate, 25  
summary.tsdistribution.profile, 26  
summary.tsdistribution.spdestimate, 27

tsmoments  
    (tsmoments.tsdistribution.estimate),  
    27  
tsmoments.tsdistribution.estimate, 27  
tsprofile  
    (tsprofile.tsdistribution.spec),  
    28  
tsprofile.tsdistribution.spec, 28

vcov, 26  
vcov (vcov.tsdistribution.estimate), 29  
vcov.tsdistribution.estimate, 29