

Package ‘gofedf’

October 1, 2023

Title Goodness of Fit Tests Based on Empirical Distribution Functions

Version 0.1.0

Description Routines that allow the user to run goodness of fit tests based on empirical distribution functions for formal model evaluation in a general likelihood model. In addition, functions are provided to test a sample against Normal or Gamma distributions, validate the normality assumptions in a linear model, and examine the appropriateness of a Gamma distribution in generalized linear models with various link functions. Michael Arthur Stephens (1976) <<http://www.jstor.org/stable/2958206>>.

License GPL (>= 3)

URL <https://github.com/pnickchi/gofedf>

BugReports <https://github.com/pnickchi/gofedf/issues>

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 2.10)

Imports stats, CompQuadForm, MASS, glm2, statmod

NeedsCompilation no

Author Richard Lockhart [aut],
Payman Nickchi [aut, cre]

Maintainer Payman Nickchi <pnickchi@sfu.ca>

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Repository CRAN

Date/Publication 2023-10-01 16:30:06 UTC

R topics documented:

<code>applyGamma</code>	2
<code>applyLMNormal</code>	3

expMLE	3
expPIT	4
expScore	4
gammaFisherByHessian	5
gammaMLE	5
gammaPIT	6
gammaScore	6
glmMLE	7
glmScorePIT	7
inversegaussianMLE	8
inversegaussianPIT	8
inversegaussianScore	9
lmFisherByHessian	9
lmMLE	10
lmPIT	10
lmScore	11
normalFisherByHessian	11
normalMLE	12
normalPIT	12
normalScore	13
testExponential	13
testGamma	14
testGLMGamma	15
testLMNormal	17
testNormal	19
testYourModel	20

Index **22**

applyGamma	<i>Apply Gamma distribution to sample and compute required components for the test.</i>
------------	---

Description

Compute Maximum likelihood estimates of the parameters in Gamma distribution, Score function evaluated at the sample, and probability inverse transformed (PIT) values of sample.

Usage

```
applyGamma(x, use.rate)
```

Arguments

x	a numeric vector.
use.rate	logical. If TRUE the rate parameter is returned while estimating MLE. Otherwise the scale is returned.

Value

a list with three elements.

applyLMNormal	<i>Apply linear model and extract required components for the test</i>
---------------	--

Description

Apply linear model and extract required components for the test

Usage

```
applyLMNormal(x, y)
```

Arguments

x	is either a numeric vector or a design matrix. In the design matrix, rows indicate observations and columns presents explanatory variables.
y	is a vector of numeric values with the same number of observations or number of rows as x.

Value

a list with three elements.

expMLE	<i>Calculate MLE of rate in Exponential dist.</i>
--------	---

Description

Calculate MLE of rate in Exponential dist.

Usage

```
expMLE(x)
```

Arguments

x	a numeric vector of length n
---	------------------------------

Value

Numeric

expPIT	<i>Compute probability inverse transform values for Exponential dist.</i>
--------	---

Description

Compute probability inverse transform values for Exponential dist.

Usage

expPIT(x, theta)

Arguments

x	a numeric vector of length n
theta	a numeric vector of length one, containing MLE rate in Exponential dist.

Value

a vector of length n containing the probability inverse transformed (PIT) values

expScore	<i>Compute score function for Exponential distribution.</i>
----------	---

Description

Compute score function for Exponential distribution.

Usage

expScore(x, theta)

Arguments

x	a numeric vector of length n
theta	a numeric vector of length two, containing MLE of parameters in Exponential dist.

Value

Score matrix with n rows and two columns.

gammaFisherByHessian	<i>Compute Fisher information matrix by the negative expected value of Hessian matrix in Gamma distribution.</i>
----------------------	--

Description

Compute Fisher information matrix by the negative expected value of Hessian matrix in Gamma distribution.

Usage

```
gammaFisherByHessian(theta)
```

Arguments

theta a numeric vector of length two, containing MLE of parameters in Gamma dist

Value

Fisher information matrix for Gamma distribution

gammaMLE	<i>Compute maximum likelihood estimate of shape and scale parameter in Gamma distribution.</i>
----------	--

Description

Estimate the MLE of shape and scale parameters of the Gamma distribution using the Newton-Raphson method on the profile log-likelihood to estimate the shape parameter.

Usage

```
gammaMLE(x, ur)
```

Arguments

x a numeric vector of length n
 ur logical. If TRUE the rate parameter is returned. Otherwise the scale is returned.

Value

a vector of length two with shape and scale/rate.

gammaPIT

Compute probability inverse transform values for Gamma distribution

Description

Compute probability inverse transform values for Gamma distribution

Usage

```
gammaPIT(x, theta)
```

Arguments

x a numeric vector of length n
theta a numeric vector of length two, containing MLE of parameters in Gamma dist.

Value

a vector of length n containing the probability inverse transformed (PIT) values

gammaScore

Compute score function for Gamma distribution.

Description

Compute score function for Gamma distribution.

Usage

```
gammaScore(x, theta)
```

Arguments

x a numeric vector of length n
theta a numeric vector of length two, containing MLE of parameters in Gamma dist.

Value

Score matrix with n rows and two columns.

glmMLE	<i>Compute maximum likelihood estimates for a generalized linear model with Gamma response.</i>
--------	---

Description

Compute maximum likelihood estimates for a generalized linear model with Gamma response.

Usage

```
glmMLE(fit)
```

Arguments

`fit` is an object of class `glm` and its default value is `NULL`. If a fit of class `glm` is provided, the arguments `x`, `y`, and `l` will be ignored. We recommend using `glm2` function from `glm2` package since it provides better convergence while optimizing the likelihood to estimate coefficients of the model by IWLS method. It is required to return design matrix by `x = TRUE` in `glm` or `glm2` function. For more information on how to do this, refer to the help documentation for the `glm` or `glm2` function.

Value

a numeric vector of estimates.

glmScorePIT	<i>Compute score function for a generalized linear model with Gamma response.</i>
-------------	---

Description

Compute score function for a generalized linear model with Gamma response.

Usage

```
glmScorePIT(fit, theta)
```

Arguments

`fit` TBD
`theta` a numeric vector of length $(p+1)$, containing MLE of parameters in a linear model.

Value

Score matrix with n rows and $(p+2)$ columns.

inversegaussianMLE	<i>Compute the maximum likelihood estimate of parameters in Inverse Gaussian distribution with weighted observations.</i>
--------------------	---

Description

This function is used in [testYourModel](#) function for example purposes.

Usage

```
inversegaussianMLE(obs, ...)
```

Arguments

obs	a numeric vector of sample observations.
...	a list of additional parameters to define the likelihood.

Value

The function compute the MLE of parameters in Inverse Gaussian distribution and returns a vector of estimates. The first and second elements of the vector are MLE of the mean and shape, respectively.

inversegaussianPIT	<i>Compute the probability transformed values for a sample from Inverse Gaussian distribution.</i>
--------------------	--

Description

This function is used in [testYourModel](#) function for example purposes.

Usage

```
inversegaussianPIT(obs, ...)
```

Arguments

obs	A numeric vector of sample observations.
...	A list of additional parameters to define the likelihood.

Value

A numeric vector of probability transformed values of sample observations.

inversegaussianScore	<i>Compute the score function of the Inverse Gaussian distribution based on a sample.</i>
----------------------	---

Description

This function is used in [testYourModel](#) function for example purposes.

Usage

```
inversegaussianScore(obs, ...)
```

Arguments

obs	a numeric vector of sample observations.
...	a list of additional parameters to define the likelihood.

Value

The score matrix with n rows (number of sample observations) and 2 columns (mean and shape).

lmFisherByHessian	<i>Compute Fisher information matrix in the case of linear model with Normal residuals.</i>
-------------------	---

Description

Compute Fisher information matrix in the case of linear model with Normal residuals.

Usage

```
lmFisherByHessian(x, y, theta)
```

Arguments

x	a matrix with n rows and p columns containing the explanatory variables.
y	a numeric vector of length n containing the response variable.
theta	a numeric vector of length (p+1), containing MLE of parameters in a linear model.

Value

Fisher information matrix for linear models.

`lmMLE`*Compute maximum likelihood estimates for linear models*

Description

Compute maximum likelihood estimates for linear models

Usage

```
lmMLE(x, y)
```

Arguments

`x` a matrix with n rows and p columns containing the explanatory variables.
`y` a numeric vector of length n containing the response variable.

Value

a numeric vector of estimates.

`lmPIT`*Compute probability inverse transform values for linear models.*

Description

Compute probability inverse transform values for linear models.

Usage

```
lmPIT(x, y, theta)
```

Arguments

`x` a matrix with n rows and p columns containing the explanatory variables.
`y` a numeric vector of length n containing the response variable.
`theta` a numeric vector of length $(p+1)$, containing MLE of parameters in a linear model.

Value

a vector of length n containing the probability inverse transformed (PIT) values

lmScore *Compute score function for linear models.*

Description

Compute score function for linear models.

Usage

```
lmScore(x, y, theta)
```

Arguments

x a matrix with n rows and p columns containing the explanatory variables.
y a numeric vector of length n containing the response variable.
theta a numeric vector of length (p+1), containing MLE of parameters in a linear model.

Value

Score matrix with n rows and (p+1) columns.

normalFisherByHessian *Compute Fisher information matrix by the negative expected value of Hessian matrix in Normal distribution.*

Description

Compute Fisher information matrix by the negative expected value of Hessian matrix in Normal distribution.

Usage

```
normalFisherByHessian(theta)
```

Arguments

theta a numeric vector of length two, containing MLE of parameters in Normal dist

Value

Fisher information matrix for Normal distribution

`normalMLE`*Compute MLE estimate for Normal*

Description

Compute MLE estimate for Normal

Usage`normalMLE(x)`**Arguments**

`x` a numeric vector of length `n`

Value

a numeric vector of length two, containing MLE of parameters in Normal dist.

`normalPIT`*Compute probability inverse transform values for Normal distribution*

Description

Compute probability inverse transform values for Normal distribution

Usage`normalPIT(x, theta)`**Arguments**

`x` a numeric vector of length `n`

`theta` a numeric vector of length two, containing MLE of parameters in Normal dist.

Value

a vector of length `n` containing the probability inverse transformed (PIT) values

normalScore	<i>Compute score function for Normal dist</i>
-------------	---

Description

Compute score function for Normal dist

Usage

```
normalScore(x, theta)
```

Arguments

x	a numeric vector of length n
theta	a numeric vector of length two, containing MLE of parameters in Normal dist.

Value

Score matrix with n rows and two columns.

testExponential	<i>Apply Goodness of Fit Test for Exponential Distribution</i>
-----------------	--

Description

Performs the goodness-of-fit test based on empirical distribution function to check if an i.i.d sample follows an Exponential distribution.

Usage

```
testExponential(
  x,
  ngrid = length(x),
  gridpit = FALSE,
  hessian = FALSE,
  method = "cvm"
)
```

Arguments

x	a non-empty numeric vector of sample data.
ngrid	the number of equally spaced points to discretize the (0,1) interval for computing the covariance function.

gridpit	logical. If TRUE (the default value), the parameter ngrid is ignored and (0,1) interval is divided based on probability inverse transformed values obtained from the sample. If FALSE, the interval is divided into ngrid equally spaced points for computing the covariance function.
hessian	logical. If TRUE the Fisher information matrix is estimated by the observed Hessian Matrix based on the sample. If FALSE (the default value) the Fisher information matrix is estimated by the variance of the observed score matrix.
method	a character string indicating which goodness-of-fit statistic is to be computed. The default value is 'cvm' for the Cramer-von-Mises statistic. Other options include 'ad' for the Anderson-Darling statistic, and 'both' to compute both cvm and ad.

Value

A list of two containing the following components:

- Statistic: the value of goodness-of-fit statistic.
- p-value: the approximate p-value for the goodness-of-fit test based on empirical distribution function. if method = 'cvm' or method = 'ad', it returns a numeric value for the statistic and p-value. If method = 'both', it returns a numeric vector with two elements and one for each statistic.

Examples

```
set.seed(123)
n <- 50
sim_data <- rexp(n, rate = 2)
testExponential(x = sim_data)
```

testGamma

Apply Goodness of Fit Test for Gamma Distribution

Description

Performs the goodness-of-fit test based on empirical distribution function to check if an i.i.d sample follows a Gamma distribution.

Usage

```
testGamma(
  x,
  ngrid = length(x),
  gridpit = FALSE,
  hessian = FALSE,
  rate = TRUE,
  method = "cvm"
)
```

Arguments

x	a non-empty numeric vector of sample data.
ngrid	the number of equally spaced points to discretize the (0,1) interval for computing the covariance function.
gridpit	logical. If TRUE (the default value), the parameter ngrid is ignored and (0,1) interval is divided based on probability inverse transformed values obtained from the sample. If FALSE, the interval is divided into ngrid equally spaced points for computing the covariance function.
hessian	logical. If TRUE the Fisher information matrix is estimated by the observed Hessian Matrix based on the sample. If FALSE (the default value) the Fisher information matrix is estimated by the variance of the observed score matrix.
rate	logical. If TRUE (the default value), the rate is estimated in Gamma distribution. If FALSE, scale is estimated. See GammaDist for more details.
method	a character string indicating which goodness-of-fit statistic is to be computed. The default value is 'cvm' for the Cramer-von-Mises statistic. Other options include 'ad' for the Anderson-Darling statistic, and 'both' to compute both cvm and ad.

Value

A list of two containing the following components:

- Statistic: the value of goodness-of-fit statistic.
- p-value: the approximate p-value for the goodness-of-fit test based on empirical distribution function. if method = 'cvm' or method = 'ad', it returns a numeric value for the statistic and p-value. If method = 'both', it returns a numeric vector with two elements and one for each statistic.

Examples

```
set.seed(123)
sim_data <- rgamma(n = 50, shape = 3)
testGamma(x = sim_data)
sim_data <- runif(n = 50)
testGamma(x = sim_data)
```

testGLMGamma

Apply Goodness of Fit Test to the Residuals of a Generalized Linear Model with Gamma Link Function

Description

testGLMGamma is used to check the validity of Gamma assumption for the response variable when fitting generalized linear model. Common link functions in [glm](#) can be used here.

Usage

```
testGLMGamma(
  x,
  y,
  fit = NULL,
  l = "log",
  hessian = FALSE,
  start.value = NULL,
  control = NULL,
  method = "cvm"
)
```

Arguments

<code>x</code>	is either a numeric vector or a design matrix. In the design matrix, rows indicate observations and columns presents covariats.
<code>y</code>	is a vector of numeric values with the same number of observations or number of rows as <code>x</code> .
<code>fit</code>	is an object of class <code>glm</code> and its default value is <code>NULL</code> . If a fit of class <code>glm</code> is provided, the arguments <code>x</code> , <code>y</code> , and <code>l</code> will be ignored. We recommend using <code>glm2</code> function from <code>glm2</code> package since it provides better convergence while optimizing the likelihood to estimate coefficients of the model by IWLS method. It is required to return design matrix by <code>x = TRUE</code> in <code>glm</code> or <code>glm2</code> function. For more information on how to do this, refer to the help documentation for the <code>glm</code> or <code>glm2</code> function.
<code>l</code>	a character vector indicating the link function that should be used for Gamma family. Some common link functions for Gamma family are 'log' and 'inverse'. For more details see <code>make.link</code> from <code>stats</code> package in R.
<code>hessian</code>	logical. If <code>TRUE</code> the Fisher information matrix is estimated by the observed Hessian Matrix based on the sample. If <code>FALSE</code> (the default value) the Fisher information matrix is estimated by the variance of the observed score matrix.
<code>start.value</code>	a numeric value or vector. This is the same as <code>start</code> argument in <code>glm</code> or <code>glm2</code> . The value is a starting point in iteratively reweighted least squares (IRLS) algorithm for estimating the MLE of coefficients in the model.
<code>control</code>	a list of parameters to control the fitting process in <code>glm</code> or <code>glm2</code> function. For more details, see <code>glm.control</code> .
<code>method</code>	a character string indicating which goodness-of-fit statistic is to be computed. The default value is 'cvm' for the Cramer-von-Mises statistic. Other options include 'ad' for the Anderson-Darling statistic, and 'both' to compute both <code>cvm</code> and <code>ad</code> .

Value

A list of three containing the following components:

- `Statistic`: the value of goodness-of-fit statistic.

- p-value: the approximate p-value for the goodness-of-fit test based on empirical distribution function. if method = 'cvm' or method = 'ad', it returns a numeric value for the statistic and p-value. If method = 'both', it returns a numeric vector with two elements and one for each statistic.
- converged: logical to indicate if the IWLS algorithm have converged or not.

Examples

```
set.seed(123)
n <- 50
p <- 5
x <- matrix( rnorm(n*p, mean = 10, sd = 0.1), nrow = n, ncol = p)
b <- runif(p)
e <- rgamma(n, shape = 3)
y <- exp(x %*% b) * e
testGLMGamma(x, y, l = 'log')
myfit <- glm(y ~ x, family = Gamma('log'), x = TRUE, y = TRUE)
testGLMGamma(fit = myfit)
```

testLMNormal

Apply Goodness of Fit Test to Residuals of a Linear Model

Description

testLMNormal is used to check the normality assumption of residuals in a linear model. This function can take the response variable and design matrix, fit a linear model, and apply the goodness-of-fit test. Conveniently, it can take an object of class "lm" and directly applies the goodness-of-fit test. The function returns a goodness-of-fit statistic along with an approximate pvalue.

Usage

```
testLMNormal(
  x,
  y,
  fit = NULL,
  ngrid = length(y),
  gridpit = FALSE,
  hessian = FALSE,
  method = "cvm"
)
```

Arguments

- x is either a numeric vector or a design matrix. In the design matrix, rows indicate observations and columns presents covariats.
- y is a vector of numeric values with the same number of observations or number of rows as x.

fit	an object of class "lm" returned by <code>lm</code> function in <code>stats</code> package. The default value of fit is NULL. If any object is provided, x and y will be ignored and the class of object is checked. If you pass an object to <code>fit</code> make sure to return the design matrix by setting <code>x = TRUE</code> and the response variable by setting <code>y = TRUE</code> in <code>lm</code> function. To read more about this see the help documentation for <code>lm</code> function or see the example below.
ngrid	the number of equally spaced points to discretize the (0,1) interval for computing the covariance function.
gridpit	logical. If TRUE (the default value), the parameter ngrid is ignored and (0,1) interval is divided based on probability inverse transformed values obtained from the sample. If FALSE, the interval is divided into ngrid equally spaced points for computing the covariance function.
hessian	logical. If TRUE the Fisher information matrix is estimated by the observed Hessian Matrix based on the sample. If FALSE (the default value) the Fisher information matrix is estimated by the variance of the observed score matrix.
method	a character string indicating which goodness-of-fit statistic is to be computed. The default value is 'cvm' for the Cramer-von-Mises statistic. Other options include 'ad' for the Anderson-Darling statistic, and 'both' to compute both cvm and ad.

Value

A list of two containing the following components:

- Statistic: the value of goodness-of-fit statistic.
- p-value: the approximate p-value for the goodness-of-fit test based on empirical distribution function. if `method = 'cvm'` or `method = 'ad'`, it returns a numeric value for the statistic and p-value. If `method = 'both'`, it returns a numeric vector with two elements and one for each statistic.

Examples

```
set.seed(123)
n <- 50
p <- 5
x <- matrix( runif(n*p), nrow = n, ncol = p)
e <- rnorm(n)
b <- runif(p)
y <- x %*% b + e
testLMNormal(x, y)
# Or pass lm.fit object directly:
lm.fit <- lm(y ~ x, x = TRUE, y = TRUE)
testLMNormal(fit = lm.fit)
```

testNormal

Apply Goodness of Fit Test for Normal Distribution

Description

Performs the goodness-of-fit test based on empirical distribution function to check if an i.i.d sample follows a Normal distribution.

Usage

```
testNormal(
  x,
  ngrid = length(x),
  gridpit = TRUE,
  hessian = FALSE,
  method = "cvm"
)
```

Arguments

x	a non-empty numeric vector of sample data.
ngrid	the number of equally spaced points to discretize the (0,1) interval for computing the covariance function.
gridpit	logical. If TRUE (the default value), the parameter ngrid is ignored and (0,1) interval is divided based on probability inverse transformed values obtained from the sample. If FALSE, the interval is divided into ngrid equally spaced points for computing the covariance function.
hessian	logical. If TRUE the Fisher information matrix is estimated by the observed Hessian Matrix based on the sample. If FALSE (the default value) the Fisher information matrix is estimated by the variance of the observed score matrix.
method	a character string indicating which goodness-of-fit statistic is to be computed. The default value is 'cvm' for the Cramer-von-Mises statistic. Other options include 'ad' for the Anderson-Darling statistic, and 'both' to compute both cvm and ad.

Value

A list of two containing the following components:

- **Statistic:** the value of goodness-of-fit statistic.
- **p-value:** the approximate p-value for the goodness-of-fit test based on empirical distribution function. if method = 'cvm' or method = 'ad', it returns a numeric value for the statistic and p-value. If method = 'both', it returns a numeric vector with two elements and one for each statistic.

Examples

```

set.seed(123)
sim_data <- rnorm(n = 50)
testNormal(x = sim_data)
sim_data <- rgamma(50, shape = 3)
testNormal(x = sim_data)

```

testYourModel

Apply the Goodness of Fit Test Based on Empirical Distribution Function to Any Likelihood Model.

Description

This function applies the goodness-of-fit test based on empirical distribution function. It requires certain inputs depending on whether the model involves parameter estimation or not. If the model is known and there is no parameter estimation, the function requires the sample as a vector and the probability transformed (or pit) values of the sample. This ought to be a vector as well. If there is parameter estimation in the model, the function additionally requires the score as a matrix with n rows and p columns, where n is the sample size and p is the number of estimated parameters. The function checks if the score is zero at the estimated parameter (which is assumed to be the maximum likelihood estimate).

Usage

```

testYourModel(
  x,
  pit,
  score = NULL,
  ngrid = length(x),
  gridpit = TRUE,
  precision = 1e-09,
  method = "cvm"
)

```

Arguments

<code>x</code>	a non-empty numeric vector of sample data.
<code>pit</code>	The probability transformed (or pit) values of the sample which ought to be a numeric vector with the same size as <code>x</code> .
<code>score</code>	The default value is null and refers to no parameter estimation case. If there is parameter estimation, the score matrix must be a matrix with n rows and p columns, where n is the sample size and p is the number of estimated parameters.
<code>ngrid</code>	the number of equally spaced points to discretize the $(0,1)$ interval for computing the covariance function.

gridpit	logical. If TRUE (the default value), the parameter ngrid is ignored and (0,1) interval is divided based on probability inverse transformed values obtained from the sample. If FALSE, the interval is divided into ngrid equally spaced points for computing the covariance function.
precision	The theory behind goodness-of-fit test based on empirical distribution function (edf) works well if the MLE is indeed the root of derivative of log likelihood function. A precision of 1e-9 (default value) is used to check this. A warning message is generated if the score evaluated at MLE is not close enough to zero.
method	a character string indicating which goodness-of-fit statistic is to be computed. The default value is 'cvm' for the Cramer-von-Mises statistic. Other options include 'ad' for the Anderson-Darling statistic, and 'both' to compute both cvm and ad.

Value

A list of two containing the following components:

- Statistic: the value of goodness-of-fit statistic.
- p-value: the approximate p-value for the goodness-of-fit test based on empirical distribution function. if method = 'cvm' or method = 'ad', it returns a numeric value for the statistic and p-value. If method = 'both', it returns a numeric vector with two elements and one for each statistic.

Examples

```
# Example: Inverse Gaussian (IG) distribution with weights

# Set the seed to reproduce example.
set.seed(123)

# Set the sample size
n <- 50

# Assign weights
weights <- rep(1.5,n)

# Set mean and shape parameters for IG distribution.
mio      <- 2
lambda  <- 2

# Generate a random sample from IG distribution with weighted shape.
sim_data <- statmod::rinvgauss(n, mean = mio, shape = lambda * weights)

# Compute MLE of parameters, score matrix, and pit values.
theta_hat <- inversegaussianMLE(obs = sim_data, w = weights)
ScoreMatrix <- inversegaussianScore(obs = sim_data, w = weights, mle = theta_hat)
pitvalues <- inversegaussianPIT(obs = sim_data, w = weights, mle = theta_hat)

# Apply the goodness-of-fit test.
testYourModel(x = sim_data, pit = pitvalues, score = ScoreMatrix)
```

Index

applyGamma, 2
applyLMNormal, 3

expMLE, 3
expPIT, 4
expScore, 4

GammaDist, 15
gammaFisherByHessian, 5
gammaMLE, 5
gammaPIT, 6
gammaScore, 6
glm, 7, 15, 16
glm.control, 16
glm2, 7, 16
glmMLE, 7
glmScorePIT, 7

inversegaussianMLE, 8
inversegaussianPIT, 8
inversegaussianScore, 9

lm, 18
lmFisherByHessian, 9
lmMLE, 10
lmPIT, 10
lmScore, 11

make.link, 16

normalFisherByHessian, 11
normalMLE, 12
normalPIT, 12
normalScore, 13

stats, 18

testExponential, 13
testGamma, 14
testGLMGamma, 15
testLMNormal, 17
testNormal, 19
testYourModel, 8, 9, 20