Package 'distfreereg'

May 8, 2026

Type Package

Title Distribution-Free Goodness-of-Fit Testing for Regression

Version 1.2

Depends R (>= 4.4)

Imports calculus, clue, lme4, methods

Suggests knitr, rmarkdown

VignetteBuilder knitr, rmarkdown

Date 2026-05-05

Maintainer Jesse Miller <smallepsilon@proton.me>

Description Implements the distribution-free goodness-of-fit regression testing procedure, introduced by Estate Khmaladze (2021, <doi:10.1007/s10463-021-00786-3>) to test whether or not the mean structure of a parametric model belongs to a specified model family. The test is implemented for general mean functions with minimal distributional assumptions as well as common models (e.g., lm, glm) with the usual model assumptions.

License GPL-3

NeedsCompilation no

Author Jesse Miller [aut, cre] (ORCID: <https://orcid.org/0009-0005-9465-7461>)

Repository CRAN

Date/Publication 2026-05-08 08:00:13 UTC

Contents

`distfreereg-package` *Distribution-Free Goodness-of-Fit Testing for Regression*

Description

Implements the distribution-free goodness-of-fit regression testing procedure, introduced by Estate Khmaladze (2021, <doi:10.1007/s10463-021-00786-3>) to test whether or not the mean structure of a parametric model belongs to a specified model family. The test is implemented for general mean functions with minimal distributional assumptions as well as common models (e.g., lm, glm) with the usual model assumptions.

Details

The DESCRIPTION file:

Package:	distfreereg
Type:	Package
Title:	Distribution-Free Goodness-of-Fit Testing for Regression
Version:	1.2
Depends:	R (>= 4.4)
Imports:	calculus, clue, lme4, methods
Suggests:	knitr, rmarkdown
VignetteBuilder:	knitr, rmarkdown
Date:	2026-05-05
Authors@R:	person(given = "Jesse", family = "Miller", role = c("aut", "cre"), email = "smallepsilon@proton.me", comm
Maintainer:	Jesse Miller <smallepsilon@proton.me>
Description:	Implements the distribution-free goodness-of-fit regression testing procedure, introduced by Estate Khmala
License:	GPL-3
Author:	Jesse Miller [aut, cre] (ORCID: <https://orcid.org/0009-0005-9465-7461>)

Index of help topics:

asymptotics	Convenience Function for Exploring Asymptotic Behavior and Sample Size Adequacy
coef.distfreereg	Extract Estimated Parameters from 'distfreereg' Objects
compare	Compare the Distributions of Empirical and Theoretical Statistics Used in Distribution-Free Parametric Regression Testing
confint.distfreereg	Calculate Confidence Intervals with a 'distfreereg' Object
distfreereg	Distribution-Free Parametric Regression Testing
distfreereg-package	Distribution-Free Goodness-of-Fit Testing for Regression
fitted.distfreereg	Extract Fitted Values from 'distfreereg' Objects
formula.distfreereg	Extract Formulas from 'distfreereg' Objects
ks.test.compare	Formally Compare Empirical and Theoretical Statistics from a 'compare' Object
plot.compare	Summary and Diagnostic Plots for 'compare' Objects
plot.distfreereg	Summary and Diagnostic Plots for 'distfreereg' Objects
predict.distfreereg	Generate Predicted Values from 'distfreereg' Objects
print.compare	Printing 'compare' Objects
print.distfreereg	Printing 'distfreereg' Objects
rejection	Compute Rejection Rates of a Distribution-Free Test from a 'compare' Object
residuals.distfreereg	Extract Residuals from 'distfreereg' Objects
update.distfreereg	Update 'distfreereg' Objects
vcov.distfreereg	Estimate Parameter Covariance Matrices from 'distfreereg' Objects

Further information is available in the following vignettes:

v1_introduction	An Introduction to the distfreereg Package (source)
v2_compare	Comparing Distributions with the distfreereg Package (source)
v3_plotting	Plotting with the distfreereg Package (source)
v4_parameter-estimation	Parameter Estimation with distfreereg (source)
v5_advanced-options	Advanced Options for the distfreereg Package (source)

Author(s)

Jesse Miller [aut, cre] (ORCID: <https://orcid.org/0009-0005-9465-7461>)

Maintainer: Jesse Miller <smallepsilon@proton.me>

`asymptotics` *Convenience Function for Exploring Asymptotic Behavior and Sample Size Adequacy*

Description

This is a convenience function that calls `compare` using `object` to create the true and test model specifications.

Usage

```
asymptotics(object, ...)
```

Arguments

`object`	Object of class `distfreereg`.
`...`	Additional arguments to pass to `compare`.

Details

An important step in implementing `distfreereg` is determining the plausibility that the sample size of a data set is adequate to produce the desired asymptotic behavior of the simulated statistics. This can be done by calling `compare` with the appropriate argument values. Because of the importance of this step, this convenience function automates that call using a single argument, namely the object of class `distfreereg` in question.

Value

Object of class `compare`.

Warning

The fitted model in `object` is assumed to be the true model, not an estimation of the true model. Further, the simulation of values requires assuming a particular error distribution. See the "Warnings" section in `compare`'s documentation for details.

Author(s)

Jesse Miller

See Also

`distfreereg`, `compare`

coef.distfreereg *Extract Estimated Parameters from* distfreereg *Objects*

Description

This is a coef method for objects of class distfreereg. It extracts the estimated parameters from a model in a distfreereg object.

Usage

```
## S3 method for class 'distfreereg'
coef(object, ...)
```

Arguments

object	Object of class distfreereg.
...	Additional parameters passed to or from other methods. Currently ignored.

Value

Numeric vector of estimated model parameters.

Author(s)

Jesse Miller

See Also

distfreereg, vcov.distfreereg, confint.distfreereg

compare *Compare the Distributions of Empirical and Theoretical Statistics Used in Distribution-Free Parametric Regression Testing*

Description

Simulate response data repeatedly with true_mean as the mean and true_covariance as the covariance structure, each time running distfreereg on the simulated data. The observed statistics and p-values are saved, as are the simulated statistics from the first replication.

This function is intended to facilitate exploring sample size adequacy and the tests' powers. For a convenient wrapper to expedite investigating sample size adequacy using a distfreereg object, see asymptotics.

See the Comparing Distributions with the distfreereg Package vignette for an introduction.

Usage

```
compare(true_mean, true_method = NULL, true_method_args = NULL, true_covariance,
true_X = NULL, true_data = NULL, theta = NULL, n = NULL, reps = 1e3,
prog = reps/10, simulate_args = NULL, err_dist_fun = NULL,
err_dist_args = NULL, keep = NULL, manual = NULL, update_args = NULL, ...)
```

Arguments

true_mean	Object specifying the mean structure of the true model. It is used to generate the true values of Y that are passed internally to distfreereg.
true_method	Character vector of length one; specifies the function (e.g., lm) to use to create a model when true_mean is a formula; should be NULL otherwise.
true_method_args	Optional list; values are passed to the function specified by true_method. Should be NULL when true_mean is not a formula.
true_covariance	Named list; specifies the covariance structures of the true error distribution in the format described in the documentation for the covariance argument of distfreereg. Required when true_mean is a function or nls object, or true_method is "nls"; should be NULL otherwise.
true_X, true_data	Optional numeric matrix or data frame, respectively; specifies the covariate values for the true model. true_X is used when true_mean is a function that has an X or x argument, and the data argument is used when true_mean is a formula or model object. At most one of these should be non-NULL. Both can be NULL when true_mean is a constant function (that is, it has no covariate argument).
theta	Numeric vector; used as the (true) parameter values for the model when true_mean is a function; should be NULL otherwise.
n	Optional integer; indicates how long each simulated data vector should be. Required only when no covariate values are specified for either the true or test mean; should be NULL otherwise. Silently converted to integer if numeric.
reps	Integer; specifies number of replications. Silently converted to integer if numeric.
prog	Integer or Inf; if finite, a progress message is given when the current repetition is a multiple of prog. Default value is reps/10, unless reps is less than 10, in which case the default is 1. If Inf, no progress messages are given. Silently converted to integer if finite numeric.
simulate_args	Optional list; specifies additional named arguments to pass to simulate.
err_dist_fun	Character string; specifies the name of the function to be used to simulate errors when true_mean is a function or nls object, or true_method is "nls". See details.
err_dist_args	Optional list; specifies additional named arguments to pass to err_dist_fun.
keep	A vector of integers, or the character string "all". If not NULL, then the output of each replication's call to distfreereg is included in the output if its repetition number is included in keep. Using keep = "all" is equivalent to keep = 1:reps.

manual	Optional function; applied to the distfreereg object created in each iteration, whose output is saved in the list manual in the output.
update_args	Optional named list; specifies arguments to pass to update.distfreereg.
...	Additional arguments passed to distfreereg. See details.

Details

This function allows the user to explore the asymptotic behavior of the distributions involved in the test conducted by distfreereg. If the sample size is large enough and the true covariance matrix of the errors is known or is estimated well enough, then the observed and simulated statistics have nearly the same distribution. How large the sample size must be depends on the details of the situation. This function can be used to determine how large the sample size must be to obtain approximately equal distributions, and to estimate the power of the test against a specific alternative.

The user specifies a particular true model which is used to generate outcome values. There are three cases:

- When true_mean is a function, this function determines the mean of the outcome values and err_dist_fun is used to generate errors. The error-generating function will usually include an element of true_covariance as an argument, and in that case must accept the appropriate class of object. For example, if the true covariance is a list of matrices corresponding to a block-diagonal covariance matrix, then err_dist_fun must accept such a list as an argument.
- When true_mean is an nls object, or when it is a formula and true_method is "nls", the function determined by the formula (in the model call or user-specified, respectively) is used to determine the mean function, and err_dist_fun generates the errors.
- When true_mean is a model object that is not an nls object, or a formula and method is not "nls", then simulate is used to generate outcome values.

If none of these cases apply to true_mean, then compare() cannot be used. (E.g., true_mean cannot be a glm object fitted using a "quasi" family, because simulate does not work for that family.)

The user also specifies arguments to pass to distfreereg, most notably a model to test comprising a mean function test_mean and a covariance structure specified by covariance. For each repetition, compare sends the simulated data, as Y or as part of data, to distfreereg.

The true_covariance argument specifies the covariance structure that is available to err_dist_fun for generating errors. The needs of err_dist_fun can vary (for example, the default function uses SqrtSigma to generate multivariate normal errors), so any one of the elements Sigma, SqrtSigma, P, and Q (defined in the documentation of distfreereg) can be specified. Any element needed by err_dist_fun is calculated automatically if not supplied.

The value of err_dist_fun must be a function whose output is a numeric matrix with n rows and reps columns. Each column is used as the vector of errors in one repetition. The error function's arguments can include the special values n, reps, Sigma, SqrtSigma, P, and Q. These arguments are automatically assigned their corresponding values from the values passed to compare. For example, the default value rmvnorm uses SqrtSigma to generate multivariate normal values with mean 0 and covariance Sigma.

The argument keep is useful for diagnosing problems, but caution should be used lest a very large object be created. It is often sufficient to save the distfreereg objects from only the first few replications.

For more specialized needs, the manual argument allows the calculation and saving of objects during each repetition. For example, using manual = function(x) residuals(x) will save the (raw) residuals from each repetition.

The first repetition creates a distfreereg object. During each subsequent repetition, this object is passed to update.distfreereg to create a new object. The update_args argument can be used to modify this call.

If necessary, global_override can be used to pass an override argument to distfreereg in each repetition. For example, using gobal_override = list(theta_hat = theta) forces the estimated parameter vector used in the test in each call to be the true parameter vector theta.

Value

An object of class compare with the following components:

call	The matched call.
Y	The matrix whose columns contain the model outcome values used for the corresponding repetitions.
theta	Supplied vector of parameter values.
true_mean	Supplied object specifying the true mean function.
true_covariance	List containing element(s) that specify the true covariance structure.
true_X	Supplied matrix of true covariate values.
true_data	Supplied data frame of true covariate values.
test_mean	Supplied object specifying the mean function being tested.
covariance	List containing element(s) that specify the test covariance structure.
X	Supplied matrix of test covariate values.
data	Supplied data frame of test covariate values.
empirical_stats	The statistics observed in each repetition.
theoretical_stats	The simulated statistics from the first repetition. (They are the same for each repetition, because compare uses update.distfreereg.)
p	The p-values for the observed statistics.
dfrs	A list containing the outputs of distfreereg for repetitions specified in keep. Included when keep is not NULL.
manual	A list containing the results of the function specified by the argument manual. Included when manual is not NULL.

Warnings

The generation of new outcome values requires specifying an error distribution. The default behavior when true_mean is a function, an nls object, or a formula with method equal to "nls" is to use a multivariate normal error distribution, but different error-generating functions can be defined by the user. When true_mean is a model object that is not an nls object, or a formula and method

is not "nls", then the errors are generated using simulate and are therefore distributed according to that function's specifications. In short, the asymptotic behavior is determined for a specific (true) error distribution, even though the test itself is distribution-free.

Simulated outcome values for model using a subset argument can result in unexpected behavior when using a subsetting condition that involves the outcome variable. Such subsetting is highly discouraged, since the subsetting will occur after outcome values have been replaced and therefore different subsets might be selected for each repetition. Further, because of the way that a data set is retrieved from an nls object, subsetting with nls using "self-referential" conditions can cause unexpected results. For example, if a condition such as x > median(x) is used, then the median will be calculated on each subset in turn, which will shrink the sample size with each repetition and throw an error.

Note

Some of the processing of the elements of true_covariance is analogous to the processing of covariance by distfreereg. Any values of solve_tol and symmetric specified in distfreereg's control argument are used by compare to similar effect in processing true_covariance.

Support for glm objects is limited to those created using a family that has a simulate element.

The presence of call in the value allows a compare object to be passed to update.

Author(s)

Jesse Miller

See Also

asymptotics, distfreereg, ks.test.compare, plot.compare, rejection

Examples

```
set.seed(20240201)
n <- 100
func <- function(X, theta) theta[1] + theta[2]*X[,1]
Sig <- rWishart(1, df = n, Sigma = diag(n))[,,1]
theta <- c(2,5)
X <- matrix(rexp(n, rate = 1))
# In practice, 'reps' should be much larger
cdfr <- compare(true_mean = func, true_X = X, true_covariance = list(Sigma = Sig),
                test_mean = func, X = X, covariance = list(Sigma = Sig),
               reps = 10, prog = Inf, theta = theta, theta_init = rep(1, length(theta)))

cdfr$p
```

confint.distfreereg *Calculate Confidence Intervals with a* distfreereg *Object*

Description

This is a confint method for objects of class distfreereg. It calculates confidence intervals for the estimated parameters of a model in a distfreereg object.

Usage

```
## S3 method for class 'distfreereg'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	Object of class distfreereg.
parm	Numeric or character vector; specifies which parameters are to be given confidence intervals. If missing, all parameters are considered.
level	Numeric vector of length one; specifies the confidence level.
...	Additional parameters passed to other methods. Currently ignored.

Details

When object contains a model object (either in the test_mean or model element), then this model is sent to confint. Otherwise, object is sent to confint.default.

Value

The output from the appropriate confint method.

Note

If object was created by calling distfreereg with no test mean function (that is, with test_mean equal to NULL), there is no estimated parameter vector, and therefore this function does not apply.

Author(s)

Jesse Miller

See Also

distfreereg, vcov.distfreereg, confint

distfreereg *Distribution-Free Parametric Regression Testing*

Description

Conduct distribution-free parametric regression testing using the process introduced in *Khmaladze (2021)*. A parametric model for the conditional mean (specified by `test_mean`) is checked against the data by fitting the model, transforming the resulting residuals, and then calculating a statistic on the empirical partial sum process of the transformed residuals. The statistic's null distribution can be simulated in a straight-forward way, thereby producing a p-value.

Using f to denote the mean function being tested, the specific test has the following null and alternative hypotheses:

$$H_0\colon\ \exists\theta \in \Theta \subseteq \mathbb{R}^p \ \big|\ \mathrm{E}(Y|X) = f(X;\theta) \quad \text{against} \quad H_1\colon\ \forall\theta \in \Theta \subseteq \mathbb{R}^p \ \big|\ \mathrm{E}(Y|X) \neq f(X;\theta).$$

This assumes a known or consistently estimated covariance matrix. See the An Introduction to the `distfreereg` Package vignette for an introduction.

Usage

```
distfreereg(test_mean = NULL, covariance = NULL, Y = NULL, X = NULL,
            theta_init = NULL, data = NULL, method = NULL, method_args = NULL,
            stat = c("KS", "CvM"), B = 1e4, ordering = "simplex", group = TRUE,
            control = NULL, override = NULL, verbose = TRUE)
```

Arguments

`test_mean`	A specification of the mean function to be tested. Accepted classes are `function`, `formula`, `glm`, `lm`, `lmerMod` (from the **lme4** package), and `nls`. Can be `NULL` if `override` contains `fitted_values` and `J` elements. See Details and Warnings.
`covariance`	Named list; specifies the covariance structure of the model's error distribution. Must be 'NULL' when 'test_mean' is a model object, in which case the covariance structure is extracted automatically from the model object. Valid element names are `"Sigma"`, `"SqrtSigma"`, `"P"`, and `"Q"`, corresponding to the covariance matrix, the square root of the covariance matrix, the precision matrix, and the square root of the precision matrix, respectively. Each element must be one of the following: • a numeric matrix. • a list of numeric matrices. • a numeric vector whose length is the sample size. • a numeric vector of length 1. See Details.
`Y`	Numeric vector of observations used when `test_mean` is a `function` or `NULL`; should be `NULL` otherwise. A non-vector value is converted to a vector.

`X` Optional numeric matrix of covariates used when `test_mean` is a `function` or `NULL`; should be `NULL` otherwise. A non-matrix value is converted to a single-column matrix.

`theta_init` Numeric vector; specifies the starting parameter values passed to the optimization function used to estimate the parameter vector. Used when `test_mean` is a `function`; should be `NULL` otherwise.

`data` Data frame of covariate values used when `test_mean` is a `formula`; should be `NULL` otherwise.

`method` Character vector; specifies the function to use for fitting the model when `test_mean` is a formula; should be `NULL` otherwise. Possible values are "`glm`", "`lm`", "`lmer`", and "`nls`".

`method_args` Optional list of argument values to be passed to the function specified by the `method` argument. Should be `NULL` when `test_mean` is not a formula.

`stat` Character vector; specifies the names of the functions used to calculate the desired statistics. By default, a Kolmogorov–Smirnov statistic and a Cramer–von Mises-like statistic are calculated:

$$\text{KS} = \max_i |a_i| \qquad \text{and} \qquad \text{CvM} = \frac{1}{n}\sum_{i=1}^{n} a_i^2$$

where a_i is term i in the empirical partial sum process and n is the sample size.

`B` Numeric vector of length one; specifies the Monte Carlo sample size used when simulating statistics. Silently converted to integer.

`ordering` A character string or a list; specifies how to order the residuals to form the empirical partial sum process. Valid character strings are:

- "`asis`": leaves the order unchanged (that is, in the order in which the observations appear in the supplied data).
- "`natural`": orders residuals using column-wise ordering of the covariates.
- "`optimal`": orders the residuals by ordering the observations using optimal transport on the covariates. The solution is estimated using the Hungarian method as implemented by `solve_LSAP`. This option can be very slow for large sets of covariates.
- "`simplex`": (the default) orders residuals in order of increasing row sums of the covariates after each column has been scaled to the interval $[0, 1]$.

If `ordering` is a list, then its elements specify names columns of `X` or `data` to use to determine the order.

`group` Logical; if `TRUE`, then residuals with the same rank (as determined by `ordering`) are grouped when calculating the empirical partial sum process.

`control` Optional named list of elements that control the details of the algorithm's computations. The following elements are accepted for all methods:

- `symmetric`: Optional named list or `FALSE`; if a named list, its elements are passed as arguments to `isSymmetric` when testing elements of `covariance` for symmetry. If `FALSE`, then this test is skipped.
- `matsqrt_tol`: Numeric; specifies the threshold for considering an eigenvalue "too negative" when calculating the square root of a matrix. Must be non-positive. The default value is `-.Machine$double.eps^0.25`.

- `solve_tol`: Numeric; passed as `tol` argument to `solve`, used in particular to invert `Sigma`. The default value is `.Machine$double.eps`.
- `qr_tol`: Numeric; passed as `tol` argument to `qr`. The default value is `sqrt(.Machine$double.eps)`. This value might need to be decreased when the dimensions of the Jacobian are sufficiently large.
- `orth_tol`: Numeric; passed as `tolerance` argument to `all.equal` when testing whether or not $r^T r$ and $\mu^T \mu$ are the identity matrix. The default value is `sqrt(.Machine$double.eps)`.
- `trans_tol`: Numeric; passed as `tolerance` argument to `all.equal` to internal transformation function when determining whether the normalizing scalar is non-zero. The default value is `sqrt(.Machine$double.eps)`.
- `sym_tol, sym_tol1`: Numeric; passed as `tol` and `tol1` arguments, respectively, to `isSymmetric` for checking `Sigma` and `P` elements of `covariance`. The default values are `sqrt(.Machine$double.eps)` and `8*sqrt(.Machine$double.eps)`, respectively. See Warnings.
- `return_on_error`: Logical; determines whether or not partial output is returned when an error is encountered. Default value is `TRUE`.

The following named elements, all but the first of which control the process of calculating the generalized least squares estimation of the parameter vector, are accepted for the `function` method:

- `jacobian_args`: Optional list; specifies arguments to pass to `jacobian`.
- `optimization_fun`: Optional function; specifies the function used to estimate the parameters. If not specified, `optim` is used with `method = "BFGS"`.
- `fun_to_optimize_arg`: Optional character string, required when `optimization_fun` is specified; specifies the name of the argument of `optimization_fun` that is assigned the function to optimize. For example, `optim` uses `"fn"`.
- `theta_init_arg`: Optional character string, required when `optimization_fun` is specified; specifies the name of the argument of `optimization_fun` that is assigned the initial parameter values for optimization. For example, `optim` uses `"par"`.
- `theta_hat_name`: Optional character string, required when `optimization_fun` is specified; specifies the name of the element of the output of `optimization_fun` that contains the estimated parameters. For example, `optim` uses `"par"` (the same character string that specifies the argument containing the initial values). See Warnings.
- `optimization_args`: Optional list; specifies additional arguments to pass to `optimization_fun`.

`override` Optional named list of arguments that override internally calculated values. Used primarily by `update.distfreereg`, but can be accessed directly. The following named elements are accepted:

- `J`: Numeric matrix. Overrides the calculation of the Jacobian.
- `fitted_values`: Numeric vector. Overrides the calculation of the fitted values.
- `res_order`: Integer vector specifying the ordering used to order the residuals for the computation of the empirical partial sum process. Overrides `ordering`. See Details.

- `theta_hat`: Numeric vector. Used as the estimated parameter vector when `test_mean` has class `function`, overriding internal computation that would have used `optimization_fun`.
- `r`: Orthogonal (numeric) matrix. Overrides the construction of the transformation anchor matrix.
- `theoretical_stats`: List. Overrides the creation of the list of simulated statistics.

`verbose` Logical; if `TRUE`, progress messages are printed. Default value is `TRUE`.

Details

This function implements distribution-free parametric regression testing. The model is specified by a mean structure and a covariance structure.

The mean structure is specified by the argument `test_mean`. This can be an object of class `function`, `formula`, `glm`, `lm`, `lmerMod`, or `nls`. It can also be `NULL`.

If `test_mean` is a function, then it must have one or two arguments: either `theta` only, or `theta` and either `X` (uppercase) or `x` (lowercase). An uppercase `X` is interpreted in the function definition as a matrix, while a lowercase `x` is interpreted as a vector. (See examples and this vignette.) The primary reason to use a lowercase `x` is to allow for a function definition using an `R` function that is not vectorized. In general, an uppercase `X` should be preferred for speed.

If `test_mean` is a formula, then it must be a formula that can be passed to `glm`, `lm`, `lmer`, or `nls`, and the `data` argument must be specified. The appropriate model is created and is then sent to `distfreereg()` for method dispatch.

The `function` method estimates parameter values, and then uses those to evaluate the Jacobian of the mean function and to calculate fitted values.

All of these methods create a Jacobian matrix and a vector of fitted values. These, along with the covariance structure, are sent the default method. The default method also allows the user to implement the algorithm even when the mean structure is not specified in `R`, as long as the Jacobian, fitted values, and covariance structure can be imported. (This is useful if a particularly complicated function is defined in another language and cannot easily be copied into `R`.)

The covariance structure for $Y|X$ must be specified using the `covariance` argument for the `function` and default methods. For other methods, the covariance is estimated automatically.

Any element of `covariance` can be a numeric matrix, a list of numeric matrices, or a numeric vector. If it is a vector, its length must be either 1 or the sample size. This option is mathematically equivalent to setting a covariance list element to a diagonal matrix with the specified value(s) along the diagonal. Using vectors, when possible, is more efficient than using the corresponding matrix. If an element of `covariance` is a list of numeric matrices, then these matrices are interpreted as the blocks of a block diagonal matrix.

Internally, `distfreereg()` only needs `Q`, so some efficiency can be gained by supplying that directly when available. When `Q` is not specified, it is calculated using whichever element is specified. When more than one of the other elements are specified, no verification of compatibility of the elements is done. `Q` is calculated using the supplied element(s) by preferring one-step operations and operations on the square-root level. (For example, if both `P` and `SqrtSigma` elements are supplied, then `Q` is calculated using `SqrtSigma` only. If `Sigma` and `P` are supplied, then `Q` is calculated using `P` only.)

The `override` argument is used primarily by `update.distfreereg` to avoid unnecessary and potentially computationally expensive recomputation. This `update` method imports appropriate values

automatically from a previously created object of class `distfreereg`, and therefore validation is not always done. Use manually with caution.

The `res_order` element of `override` must be a vector of integers from 1 to n (the sample size) that determines the order of the residuals to use when forming the empirical partial sum process. Elements of the vector can be repeated, in which case the residuals corresponding to matching `res_order` values are grouped when `group` is `TRUE`.

Value

An object of class `distfreereg` with the following components:

`call`	The matched call.
`data`	A list containing the `data` argument value, if present, and `Y` and `X`.
`test_mean`	The value supplied to the argument `test_mean`.
`model`	The model built when using the `formula` method; present if and only if using that method.
`covariance`	The list of covariance matrices, containing at least `Q`.
`theta_hat`	The estimated parameter vector. When the model being tested has class `lmerMod`, this is the vector of fixed effect parameters as returned by `fixef`.
`optimization_output`	The output of `optimization_fun` or `nls` from calculating `theta_hat`.
`fitted_values`	The vector of fitted values, $f\left(X;\hat{\theta}\right)$. When testing a generalized linear model, these are the fitted values on the response scale.
`J`	The Jacobian matrix, $\frac{\partial f}{\partial \theta}(X;\hat{\theta})$.
`mu`	The mu matrix.
`r`	The matrix of transformation anchor vectors.
`r_tilde`	The matrix of modified transformation anchor vectors.
`residuals`	A named list of three vectors containing raw, sphered, and transformed residuals.
`res_order`	A numeric vector indicating the ranking of the residuals used to form the empirical partial sum process, in a format to be used as the input of `order`.
`grouping_matrix`	The matrix used to group residuals; present if `group` is `TRUE` and some covariate combinations are repeated.
`epsp`	The empirical partial sum process formed by calculating the scaled partial sums of the transformed residuals ordered according to `res_order`.
`observed_stats`	A named list of the observed statistic(s) corresponding to the transformed residuals.
`theoretical_stats`	A named list, each element of which contains the values of a simulated statistic.
`p`	A named list with two elements: `value`, which contains the p-values for each observed statistic, and `mcse`, which contains the Monte Carlo standard errors for the p-values.

Warnings

Methods for model objects (e.g., `lm` objects) are intended to be used with objects created using a `data` argument that contains all variables used by the model. This data argument is assumed to be defined in the same environment as the model object, and this is assumed to be the environment in which `distfreereg()` is called.

Consistency between `test_mean` and `theta_init` is verified only indirectly. Uninformative errors can occur when, for example, `theta_init` does not have the correct length. The most common error message that arises in this case is "`f_out` cannot have `NA` `values`", which occurs when `theta_init` is too short. To be safe, always define `test_mean` to use every element of `theta`.

No verification of consistency is done when multiple elements of `coviariance` are specified. For example, if `P` and `Sigma` are both specified, then the code will use only one of these, and will not verify that `P` is the inverse of `Sigma`.

When using the `control` argument element `optimization_fun` to specify an optimization function other than `optim`, the verification that `theta_hat_name` actually matches the name of an element of the optimization function's output is done only after the optimization has been done. If this optimization will likely take a long time, it is important to verify the value of `theta_hat_name` before running `distfreereg()`.

The default values of `sym_tol` and `sym_tol1` are intended to check for substantial asymmetry such as would be caused by the user inputting the wrong matrix. Therefore, they do not check for symmetry with high precision. In particular, these default values are orders of magnitude larger than the default values in `isSymmetric`.

The theory described by *Khmaladze (2021)* does not apply directly to generalized linear models nor linear mixed-effects models, but extensive simulations indicate that the method applies nonetheless to these models. Further investigation can be done using `asymptotics`. Non-null values for the `weights` argument in `glm` are not supported.

Author(s)

Jesse Miller

References

Khmaladze, Estate V. *Distribution-free testing in linear and parametric regression*, 2021-03, Annals of the Institute of Statistical Mathematics, Vol. 73, No. 6, p. 1063–1087. doi:10.1007/s10463021-007863

See Also

`coef.distfreereg`, `confint.distfreereg`, `fitted.distfreereg`, `formula.distfreereg`, `plot.distfreereg`, `predict.distfreereg`, `print.distfreereg`, `residuals.distfreereg`, `update.distfreereg`, `vcov.distfreereg`

Examples

```
set.seed(20240218)
n <- 1e2
func <- function(X, theta) X[,1]^theta[1] + theta[2]*X[,2]
Sig <- runif(n, min = 1, max = 3)
```

```
theta <- c(2,5)
X <- matrix(runif(2*n, min = 1, max = 5), nrow = n)
Y <- X[,1]^theta[1] + theta[2]*X[,2] + rnorm(n, sd = sqrt(Sig))
(dfr <- distfreereg(Y = Y, X = X, test_mean = func,
covariance = list(Sigma = Sig),
theta_init = c(1,1)))

# Same test with lowercase "x" for reference;
# use uppercase whenever possible.
func_lower <- function(x, theta) x[1]^theta[1] + theta[2]*x[2]
(dfr_lower <- distfreereg(Y = Y, X = X, test_mean = func_lower,
covariance = list(Sigma = Sig),
theta_init = c(1,1)))
```

`fitted.distfreereg` *Extract Fitted Values from* `distfreereg` *Objects*

Description

This is a `fitted` method for objects of class `distfreereg`.

Usage

```
## S3 method for class 'distfreereg'
fitted(object, ...)
```

Arguments

`object`	Object of class `distfreereg`.
`...`	Additional parameters passed to or from other methods. Currently ignored.

Value

Numeric vector of fitted values.

Author(s)

Jesse Miller

See Also

`distfreereg`

formula.distfreereg *Extract Formulas from* distfreereg *Objects*

Description

This is a formula method for objects of class distfreereg. It extracts the formula from a model in a distfreereg object.

Usage

```
## S3 method for class 'distfreereg'
formula(x, ...)
```

Arguments

x	Object of class distfreereg.
...	Additional parameters passed to or from other methods. Currently ignored.

Value

Formula extracted from x$test_mean, or NULL if such a formula cannot be extracted.

Author(s)

Jesse Miller

See Also

distfreereg

ks.test.compare *Formally Compare Empirical and Theoretical Statistics from a* compare *Object*

Description

This is a ks.test method for objects of class compare. It performs a two-sample Kolmogorov–Smirnov test to compare the observed and simulated statistics in an object of class compare.

Usage

```
## S3 method for class 'compare'
ks.test(x, ..., stat = NULL)
```

Arguments

x	Object of class compare.
...	Additional parameters passed to ks.test.
stat	Character string specifying the statistic on which to run the test.

Details

When stat is NULL, the default value is the first statistic appearing in the observed_stats element of object.

Value

A list of the form specified in ks.test.

Author(s)

Jesse Miller

See Also

compare, distfreereg, ks.test

Examples

```
# In practice, set "reps" larger than 200.
set.seed(20240201)
n <- 100
func <- function(X, theta) theta[1] + theta[2]*X[,1]
Sig <- rWishart(1, df = n, Sigma = diag(n))[,,1]
theta <- c(2,5)
X <- matrix(rexp(n, rate = 1))
cdfr <- compare(true_mean = func, true_X = X, true_covariance = list(Sigma = Sig),
                test_mean = func, X = X, covariance = list(Sigma = Sig), reps = 200,
                prog = Inf, theta = theta, theta_init = rep(1, length(theta)))

ks.test(cdfr)
ks.test(cdfr, stat = "CvM")
```

---

plot.compare *Summary and Diagnostic Plots for* compare *Objects*

Description

This is a plot method for objects of class compare. It automates the creation of four summary and diagnostic plots for compare objects. See the Plotting with the distfreereg Package vignette for examples.

Usage

```
## S3 method for class 'compare'
plot(x, y, ..., which = "cdf", stat = NULL, hlines = NULL, curve_args = NULL,
confband_args = FALSE, density_args = NULL, poly = NULL, legend = NULL,
qqline = NULL)
```

Arguments

x	Object of class compare.
y	Optional object of class compare.
...	Additional parameters passed to a plotting function depending on the value of which: to plot for "cdf" and "dens"; to qqplot for "qq" and "qqp".
which	Character string. Acceptable values are "cdf", "dens", "qq", and "qqp": • "cdf" produces a plot of the estimated cumulative distribution functions of the two vectors of statistics being compared. • "dens" produces a plot of the estimated density functions of the two vectors of statistics being compared. • "qq" produces a quantile–quantile plot comparing the two vectors of statistics. • "qqp" produces a quantile–quantile plot comparing the p-values with uniform quantiles. (This is not available when y is present.)
stat	Character string, specifies the statistic to plot.
hlines	An optional list of arguments to pass to abline, used to create the horizontal dashed lines when which is "cdf". Setting equal to FALSE prevents the call, and no lines are drawn.
curve_args	An optional list used to pass arguments to lines (not curve!), used to create the curves when which is "cdf" or "dens". It can have two special named arguments, obs and mcsim, whose values must be lists. Those lists contain arguments passed to the calls to lines for plotting the curves for the empirical and theoretical statistics, respectively. Any other elements are passed to both calls.
confband_args	An optional list of values that control the calculation and plotting of confidence bands when which is "cdf" or "dens". Any of the following named elements are allowed. • w: Numeric; the sequence of points on which to evaluate the confidence band. By default, it is a sequence of m evenly-spaced numbers whose endpoints are determined by the quantiles of x calculated using the probabilities specified by q_probs, described below. • m: Integer; the length of w, used only when w is NULL. The default value is 100. • batch_len: Integer; the batch length for the algorithm. The default value is 50. • N: Integer; the number of multivariate t samples to use in the simulation. • conf.level: Numeric; the desired confidence level. • q_probs: Numeric vector of length 2; the probabilities sent to quantile to calculate the bounds used to define w, used only when w is NULL.

- curve_args: An optional list of arguments passed to lines (again, not curve!), used to create the boundaries of the confidence band. It can have two special named arguments, obs and mcsim, which function in the same way as the corresponding elements of the curve_args argument described above.
- polygon_args: An optional list of arguments passed to polygon, used to shade the confidence region. Setting equal to FALSE prevents the call, and no shading is done.
- shade_col: This provides a shortcut to the col argument of polygon to change the color of the shaded region.

Setting equal to FALSE prevents calculation and plotting of the band.

density_args — An optional list of arguments passed to density when which is "dens", which calculates the points used to plot the density curves. The list can have two special named elements, obs and mcsim, which function in the same way as the corresponding elements of the curve_args argument described above.

poly — An optional list of arguments passed to polygon when which is "dens", which shades the area under the density curves. The list can have two special named elements, obs and mcsim, which modify the shadings for their respective curves, analogous to their behavior in the curve_args argument. When poly is equal to FALSE, no call is made, and therefore no shading is done.

legend — An optional list of arguments passed to legend when which is "cdf" or "dens". When equal to FALSE, no call is made, and therefore no legend is created.

qqline — An optional list of arguments passed to abline when which is "qq" or "qqp". By default, this plots the line $y = x$. When equal to FALSE, no call is made, and therefore no line is plotted.

Details

This function produces a plot of a type specified by which. The values plotted depend on whether or not y is present and the value of which. When y is present, the plots compare the empirical statistics in x and the empirical statistics in y. When y is missing, the plots compare the empirical and theoretical statistics in x. (The exception is when which is "qqp", which is only available when y is missing.)

When which is "cdf" or "dens", the plotting region and associated labels, tick marks, etc., are created by an initial call to plot. The curves themselves are drawn with lines. The arguments specified in ... are passed to the initial call to plot.

Value

The values used to create the curves (or points, in the case of a Q–Q plot) are returned invisibly. The details depend on the value of which:

- cdf: A list with two or four elements, all lists. The first two sub-lists contain the x- and y-values cdf curves. If confidence bands are plotted, then two additional elements are included with output from the confidence band calculations, including elements w, cb_lower, and cb_upper, which contain, respectively, the x-coordinates for both the upper and lower bounds of the band, the y-coordinates for the lower band, and the y-coordinates for the upper band.

- dens: A list with two or four elements, all lists. The first two sub-lists contain x- and y-values for the density curves. If confidence bands are plotted, then two additional sub-lists are supplied, with contents identical to what is described for "cdf".
- qq, qqp: The output of qqplot.

For "cdf" and "dens", the names of the elements of the returned list depend on whether or not a value for the argument y was supplied.

Author(s)

Jesse Miller

References

Flegal, James M. et al. **Simultaneous confidence bands for (Markov chain) Monte Carlo simulations**, forthcoming.

See Also

distfreereg, compare

---

plot.distfreereg *Summary and Diagnostic Plots for* distfreereg *Objects*

Description

This is a plot method for objects of class distfreereg. It automates the creation of summary and diagnostic plots for distfreereg objects. See the Plotting with the distfreereg Package vignette for examples.

Usage

```
## S3 method for class 'distfreereg'
plot(x, which = "dens", stat = NULL, density_args = NULL,
polygon_args = NULL, confband_args = NULL, abline_args = NULL,
shade_col = rgb(1,0,0,0.5), text_args = NULL, ...)
```

Arguments

x	Object of class distfreereg.
which	Character string. Acceptable values are "dens", "ecdf", "residuals", and "epsp": • "dens" produces a plot of the estimated density curve of the specified statistic. • "ecdf" produces a plot of the empirical cumulative distribution function of the specified statistic.

- "residuals" produces a plot of the transformed residuals in the order specified by x$res_order.
- "epsp" produces a plot of the empirical partial sum process of the (ordered) transformed residuals.

stat — Character vector of length one specifying the name of the statistic to plot when which is "dens" or "ecdf". By default, the first statistic in x$observed_stats is used.

density_args — An optional list of arguments to pass to density when which is "dens".

polygon_args — An optional list of arguments to pass to polygon, used when which is "dens" to shade under the density curve to the right of the value of the observed statistic. Setting equal to FALSE prevents the call, and no shading is done.

confband_args — An optional list of values that control the calculation and plotting of confidence bands. Any of the following named elements are allowed.

- w: Numeric; the sequence of points on which to evaluate the confidence band. By default, it is a sequence of m evenly-spaced numbers whose endpoints are determined by the quantiles of x calculated using the probabilities specified by q_probs, described below.
- m: Integer; the length of w, used only when w is NULL. The default value is 100.
- batch_len: Integer; the batch length for the algorithm. The default value is 50.
- N: Integer; the number of multivariate t samples to use in the simulation.
- conf.level: Numeric; the desired confidence level.
- q_probs: Numeric vector of length 2; the probabilities sent to quantile to calculate the bounds used to define w, used only when w is NULL.
- curve_args: An optional list of arguments passed to lines (not curve!), used to create the boundaries of the confidence band.
- polygon_args: An optional list of arguments passed to polygon, used to shade the confidence region. Setting equal to FALSE prevents the call, and no shading is done.
- shade_col: This provides a shortcut to the col argument of polygon to change the color of the shaded region.

Setting equal to FALSE prevents calculation and plotting of the confidence band.

abline_args — An optional list of arguments to pass to abline, used to draw a vertical line at the value of the observed statistic. Setting equal to FALSE prevents the call, and no line is drawn.

shade_col — Character string or other value specifying the color to use to shade the upper tail of the distribution when which is "dens". Default value is red with 50% transparency. This is a convenience argument, and the same functionality is available by defining a col element in the polygon_args argument.

text_args — An optional list of arguments to pass to text, used to label the vertical line with the p-value of the observed statistic. Setting equal to FALSE prevents the call, and no text is printed.

... — Additional arguments to pass to plot.

Details

This function produces one of three specified plots, depending on the value of `which`.

When `which` is "`dens`" or "`ecdf`", a plot of the estimated density or empirical cumulative distribution function, respectively, of the simulated statistics is produced, including a vertical line at the value of the observed test statistic with the p-value displayed.

The default placement of the p-value text is on the left side of the line indicating the statistic value. Specifically, the default values of `x` and `y` passed to `text` are the statistic value itself and the midpoint between zero and the maximum value of the density curve. The default value passed to `adj` is `c(1,0.5)`, meaning that the text is aligned to the left of the value (x, y) and centered vertically on it. (The default value for the text itself, which can be modified via the `label` argument of `text`, includes a space on the left and the right for padding so the text does not overlap the vertical line itself.) To align the text so it appears on the right side (for example, to avoid overlapping the density curve), use `text_args = list(adj = c(0,0.5))`. See documentation for `text` for details on this and other arguments.

When `which` is "`residuals`", a time-series-like plot is produced showing transformed residuals in the order given by `x$res_order`. In the case that the null hypothesis is rejected, this plot can help determine where (in terms of the linearly ordered covariates) a discrepancy between the model and the data occurs.

When `which` is "`epsp`", a plot of the empirical partial sum process is produced; that is, the y-values are

$$y_j = \frac{1}{\sqrt{n}} \sum_{i=1}^{j} \hat{e}_i$$

where $\hat{e}_i$ is the ith transformed residual in the order given by `x$res_order`. Similar to the case when `which` is "`residuals`", this plot can help determine where (in terms of the linearly ordered covariates) a discrepancy between the model and the data occurs.

Value

When `which` is "`dens`" or "`ecdf`", the values used to create the plot are returned invisibly in a list with two named elements, `x` and `y`. If the confidence band is plotted, then it is included as an element named `confband`.

For other values of `which`, nothing is returned.

Author(s)

Jesse Miller

References

Flegal, James M. et al. **Simultaneous confidence bands for (Markov chain) Monte Carlo simulations**, forthcoming.

See Also

`distfreereg`

predict.distfreereg *Generate Predicted Values from* distfreereg *Objects*

Description

This is a predict method for objects of class distfreereg.

Usage

```
## S3 method for class 'distfreereg'
predict(object, ...)
```

Arguments

object	Object of class distfreereg.
...	Additional arguments passed to other predict methods. In particular, can include a newdata value.

Details

When object$test_mean is a model object ("lm", "glm", or "nls"), object$test_mean is sent to predict for method dispatch. When object$test_mean is of class "formula", object$model is sent to predict.

Value

Numeric vector of predicted values.

Author(s)

Jesse Miller

See Also

distfreereg

print.compare *Printing* compare *Objects*

Description

This is a print method for objects of class compare.

Usage

```
## S3 method for class 'compare'
print(x, ...)
```

Arguments

x	Object of class compare.
...	Additional parameters, currently ignored.

Details

This function prints a useful summary of the compare object x.

Value

No return value (NULL).

Author(s)

Jesse Miller

See Also

compare

print.distfreereg *Printing* distfreereg *Objects*

Description

This is a print method for objects of class distfreereg.

Usage

```
## S3 method for class 'distfreereg'
print(x, ..., digits = 3, col_sep = 2, show_params = TRUE)
```

Arguments

x	Object of class distfreereg.
...	Additional parameters, currently ignored.
digits	Integer; passed to signif to determine the number of significant digits to display.
col_sep	Integer; specifies the padding (in units of spaces) between columns in the printed table of statistics.
show_params	Logical; determines whether or not the parameter estimates, if present in x, are displayed.

Details

This function prints a useful summary of the distfreereg object x.

Value

No return value (NULL).

Author(s)

Jesse Miller

See Also

distfreereg

---

rejection *Compute Rejection Rates of a Distribution-Free Test from a* compare *Object*

Description

Compute the rejection rates of the tests simulated in a compare object. Specifically, this function estimates the rejection rates of the tests conducted with specified statistics of the hypothesis that the mean function is test_mean when the true mean function is true_mean.

Usage

```
rejection(object, alpha = 0.05, stat = names(object[["empirical_stats"]]), ...)
```

Arguments

`object`	Object of class `compare`.
`alpha`	Numeric vector; specifies the α-levels to use. Passed as `probs` argument to `quantile`.
`stat`	Character vector; specifies the names of the statistics to use. The default value computes the rejection rate associated with every statistic in `object`.
`...`	Additional arguments to pass to `quantile` to estimate the $1 - \alpha$ quantiles of the distribution of simulated statistics.

Value

Data frame containing estimated rejection rates and associated Monte Carlo standard errors, with one row for each combination of `stat` and `alpha` elements.

Warning

The reported Monte Carlo standard error does not account for the uncertainty of the estimation of the $1 - \alpha$ quantiles of the distribution of simulated statistics. The number of Monte Carlo simulations should be large enough to make this estimate sufficiently accurate that it can be considered known for practical purposes. The standard errors of estimated quantiles can be calculated using the **mcmcse** package.

Author(s)

Jesse Miller

See Also

`distfreereg`, `compare`

Examples

```
# In practice, set "reps" much larger than 20.
set.seed(20240201)
n <- 100
func <- function(X, theta) theta[1] + theta[2]*X[,1]
Sig <- rWishart(1, df = n, Sigma = diag(n))[,,1]
theta <- c(2,5)
X <- matrix(rexp(n, rate = 1))
cdfr <- compare(true_mean = func, true_X = X, true_covariance = list(Sigma = Sig),
                test_mean = func, X = X, covariance = list(Sigma = Sig), reps = 20,
                prog = Inf, theta = theta, theta_init = rep(1, length(theta)))

rejection(cdfr)
rejection(cdfr, stat = "CvM")
rejection(cdfr, alpha = c(0.1, 0.2))
```

residuals.distfreereg *Extract Residuals from* distfreereg *Objects*

Description

This is a residuals method for objects of class distfreereg. It can extract any of the three available types of residuals.

Usage

```
## S3 method for class 'distfreereg'
residuals(object, ..., type = "raw")
```

Arguments

object	Object of class distfreereg.
...	Additional parameters passed to or from other methods. Currently ignored.
type	Character string specifying the type of residuals to return. Must be one of "raw", "sphered", and "transformed".

Value

Numeric vector of residuals.

Author(s)

Jesse Miller

See Also

distfreereg

update.distfreereg *Update* distfreereg *Objects*

Description

This is a distfreereg method for update. The method takes advantage of the override argument of distfreereg to prevent unnecessary recalculation of potentially computationally expensive objects.

Usage

```
## S3 method for class 'distfreereg'
update(object, ..., smart = TRUE)
```

Arguments

object	Object of class distfreereg.
...	Additional named parameters to pass to distfreereg.
smart	Logical. If TRUE, then saved values from object are passed to distfreereg using the override argument, when they need not themselves be updated. See details.

Details

This function updates an object of class distfreereg. By default, it does so "intelligently" in the sense that it does not unnecessarily recompute elements that are already saved in object. For example, if a new value for covariance is not included in ..., then the value of covariance saved in object is automatically passed to the new call, preventing recalculating Q. If a new value of covariance is specified, then all objects dependent on that (e.g., $\hat{\theta}$) are recomputed by default.

In particular, the simulated samples depend on the data and function only through the number of observations, the covariates (if any), and the dimension of the parameter space of the function. If none of these change, then the updated object reuses the simulated samples from the supplied object.

This function uses the ‘override‘ argument of distfreereg, and therefore it must be handled carefully when it itself is being updated. To create the value of ‘override‘ for the updated call, the following three named lists are combined. When a name appears in more than one of these lists, priority is given in the order in which the lists are shown:

- The value of ‘override‘ supplied to update.
- Values of ‘fitted_values‘ and ‘J‘ supplied to the ‘override‘ argument in the ‘call‘ element of ‘object‘, if the value of ‘test_mean‘ in the updated call is ‘NULL‘.
- The list of "intelligently chosen" override values determined by update.distfreereg.

Value

An updated object of class distfreereg.

Note

The usual behavior of update is to create an updated call and then evaluate that call. This is what update.distfreereg does, as well, but some of the updated elements are drawn from object itself for use as override values. In general, an object created by update.distfreereg is not identical to the object created by distfreereg using corresponding arguments, because the call values will differ.

Author(s)

Jesse Miller

See Also

distfreereg

Examples

```
set.seed(20240218)
n <- 1e2
func <- function(X, theta) X[,1]^theta[1] + theta[2]*X[,2]
Sig <- runif(n, min = 1, max = 3)
theta <- c(2,5)
X <- matrix(runif(2*n, min = 1, max = 5), nrow = n)
Y <- X[,1]^theta[1] + theta[2]*X[,2] + rnorm(n, sd = sqrt(Sig))
dfr_1 <- distfreereg(Y = Y, X = X, test_mean = func,
                     covariance = list(Sigma = Sig),
                     theta_init = c(1,1))

func_updated <- function(X, theta) X[,1]^theta[1] + theta[2]*X[,2]^2
dfr_2 <- update(dfr_1, test_mean = func_updated)
```

---

vcov.distfreereg *Estimate Parameter Covariance Matrices from* distfreereg *Objects*

Description

This is a vcov method for objects of class distfreereg. It estimates the covariance matrix of the estimated parameters in a model from a distfreereg object.

Usage

```
## S3 method for class 'distfreereg'
vcov(object, ..., jacobian_args, hessian_args)
```

Arguments

object	Object of class distfreereg.
...	Additional parameters passed to other methods when test_mean element of object is not of class function.
jacobian_args, hessian_args	Lists of additional arguments to pass to jacobian and hessian.

Details

When the test_mean element of object is of class function, the covariance matrix is estimated using the method described in section 5.3 of *Van der Vaart (2007)*. Otherwise, test_mean is of a class that has its own method for vcov, which is used to calculate the output.

Value

Named numeric matrix equal to the estimated covariance matrix of the parameter estimates from object.

Warning

This calculation can be computationally intensive when the sample size is large and `object$test_mean` is a function.

Note

If `object` was created by calling `distfreereg` with no test mean function (that is, with `test_mean` equal to `NULL`), there is no estimated parameter vector, and therefore this function does not apply.

Author(s)

Jesse Miller

References

Vaart, A. W. **Asymptotic statistics**, 2007, *Cambridge series on statistical and probabilistic mathematics*, Cambridge University Press.

See Also

`distfreereg`, `confint.distfreereg`

Index