Package 'circglmbayes'

March 9, 2018

Type Package

Date 2018-02-27

Title Bayesian Analysis of a Circular GLM

Version 1.2.3

Maintainer Kees Mulder <keestimmulder@gmail.com>

Description Perform a Bayesian analysis of a circular outcome General Linear Model (GLM), which allows regressing a circular outcome on linear and categorical predictors. Posterior samples are obtained by means of an MCMC algorithm written in 'C++' through 'Rcpp'. Estimation and credible intervals are provided, as well as hypothesis testing through Bayes Factors.
See Mulder and Klugkist (2017) <doi:10.1016/j.jmp.2017.07.001>.

License GPL-3

Encoding UTF-8

ByteCompile true

URL https://github.com/keesmulder/circglmbayes

BugReports https://github.com/keesmulder/circglmbayes/issues

LazyData true

Depends R (>= 2.10)

LinkingTo Rcpp, BH, RcppArmadillo

Imports Rcpp, stats, graphics, shiny, grDevices, ggplot2, reshape2, coda

RoxygenNote 6.0.1

NeedsCompilation yes

Author Kees Mulder [aut, cre]

Repository CRAN

Date/Publication 2018-03-09 11:58:29 UTC

R topics documented:

`arcDistance` *Compute the arc distance between two angular vectors*

Description

Compute the arc distance between two angular vectors

Usage

```
arcDistance(th1, th2)
```

Arguments

th1	The first angular vector in radians.
th2	The second angular vector in radians.

Value

The distance in radians.

BF.circGLM *Obtain Bayes Factors or posterior odds from circGLM objects*

Description

Extracts the Bayes Factors or posterior odds from a circGLM object.

Usage

```
BF.circGLM(m, prior_odds = 1, digits = 5)
```

Arguments

m	A circGLM object.
prior_odds	Numeric; If prior odds is 1, the default, the results are the Bayes factors. The priors odds can also be provided in order to return posterior odds directly, which are equal to the Bayes factor multiplied by the prior odds.
digits	Integer; The number of digits to display.

Value

A list of tables of Bayes Factors and posterior model probabilities, where applicable.

Examples

```
dat <- generateCircGLMData(truebeta = c(0, .2), truedelta = c(.4, .01))
m   <- circGLM(th ~ ., dat)
BF.circGLM(m)

dat <- generateCircGLMData(nconpred = 0)
m   <- circGLM(th ~ ., dat)
BF.circGLM(m)

dat <- generateCircGLMData(ncatpred = 0)
m   <- circGLM(th ~ ., dat)
BF.circGLM(m)
```

cglmShiny *cglmShiny*

Description

Run a shiny app interface for this package. Provides a point-and-click interface where the user can load their own data.

Usage

```
cglmShiny()
```

Examples

```
## Not run:
cglmShiny()

## End(Not run)
```

circGLM *Fitting Bayesian circular General Linear Models*

Description

The main function for running Bayesian circular GLMs. The model predicts some circular outcome θ and has the form

$$\theta_i = \beta_0 + \delta^t d_i + g(\beta^t x_i) + \epsilon_i,$$

where β_0 is an circular intercept, δ are group difference parameters, d_i is a vector of dummy variables indicating group membership, $g(.)$ is a link function given by $g(x) = ratan(x)$ where `r` can be chosen, β is a vector of regression coefficients, x_i is a vector of covariates, and ϵ_i is a von Mises distributed error with residual concentration κ. This function returns a `circGLM` object which can be further investigated with standard functions `plot`, `print`, `coef`, `residuals`, and special functions `mcmc_summary.circGLM` for results for all MCMC chains, `IC_compare.circGLM` for a comparison of information criteria of one or more circGLM models, `BF.circGLM` to obtain Bayes Factors, and `predict_function.circGLM` to create a prediction function.

Usage

```
circGLM(formula, data, th, X = if (missing(th)) {     model.matrix(formula,
  data)[, -1, drop = FALSE] } else {     matrix(nrow = length(th), ncol = 0) },
  conj_prior = rep(0, 3), bt_prior_musd = c(mu = 0, sd = 1),
  starting_values = c(0, 1, rep(0, ncol(X))), bwb = rep(0.05, ncol(X)),
  Q = 10000, burnin = 1000, thin = 1, kappaModeEstBandwith = 0.1,
```

```
  CIsize = 0.95, r = 2, returnPostSample = TRUE, output = "list",
  SDDBFDensEstMethod = "density", reparametrize = TRUE,
  groupMeanComparisons = TRUE, skipDichSplit = FALSE, centerOnly = FALSE)
```

Arguments

formula	an optional object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted.
data	an optional data frame or object coercible by as.data.frame to a data frame, containing the variables in the model.
th	An optional vector of angles in radians or degrees, representing the circular outcome we want to predict. If any value is larger than 2 * pi, the input is transformed to radians. Otherwise, th is treated as radians.
X	An optional matrix of predictors, both continuous (linear) and categorical (as dummies). If categorical predictors are included, the dummies must already be made and they must be in (0, 1), because this is checked to be able to separate them from the continuous predictors, so that they are treated differently. If not, or if *skipDichSplit = TRUE*, they will be treated as linear predictors.
conj_prior	A numeric vector of length 3, containing, in that order, prior mean direction, prior resultant length, and prior sample size. Used for the von Mises part of the model, beta_0 and kappa.
bt_prior_musd	A numeric vector of length 2, or NA. If bt_prior_musd = NA, a constant prior is used. If it is a numeric vector of length 2, a Normal prior is used so that the first value is the mean, and the second value is the standard deviation.
starting_values	A numeric vector with starting values for the MCMC sampler. The length of the numeric vector should be 2 plus the number of columns in X.
bwb	A numeric vector, where the length is at least the number of continuous predictors. This is a tuning parameters used in sampling of beta. New values are sampled uniformly around the current value of beta with bounds at bt_cur - bwb and bt_cur + bwb. If reparametrize = TRUE, bwb corresponds to the bounds around the reparametrized values.
Q	Integer; The number of iterations to perform.
burnin	Integer; The number of burn-in (warmup) iterations.
thin	Integer; The number of parameters sets to sample for each parameter set that is saved. Can be used to save memory if Q is large.
kappaModeEstBandwith	Numeric between 0 and 1. The mode of kappa is estimated by taking the midpoint of a highest density interval. Specifically, it is the midpoint of the interval that contains kappaModeEstBandwith of the density of the posterior. Reasonable values are roughly between .005 and .2, although lower values may be reasonable if Q is large.
CIsize	The size of the credible intervals. This is used for all parameters, whether they use highest density intervals, circular quantiles or regular quantiles.

r — A numeric. r is the parameter used in the link function $g(x, r) = ratan(x)$. If r = 2, the link function maps the real line to the full circle. If r < 2 the link functions maps to a proportion r / 2 of the circle. If r > 2, the link functions can reach the same are of the circle multiple times, which is unlikely to be useful, and should be used with caution.

returnPostSample — Logical indicating whether the MCMC sample itself should be returned. Should only be set to FALSE if there are memory constraints, as many subsequent analyses rely on the posterior sample directly.

output — A character string, either "list" or "vector". In most situations, "list" should be used, which returns a circGLM object. The "vector" options is only useful for simulation studies etc.

SDDBFDensEstMethod — A character string, either "density" or "histogram". Gives the method to If SDDBFDensEstMethod = "density", the default, the Bayes Factors are computed based on the density estimate given by a spline interpolation of the density() function, so they are calculated in R rather than C++. This method should be much more stable than the histogram method, especially if there is low probability at 0 in the posterior. If SDDBFDensEstMethod = "histogram", Bayes factors are computed by estimating the density from the posterior sample as the midpoint of a histogram bar at 0 containing 10% of the data.

reparametrize — Logical; If TRUE, proposals for beta are drawn uniformly around a reparametrization zt = pi * atan(bt) / 2, so from zt_can = runif(1, zt - bwb, zt + bwb), which is then transformed back. Then, the proposals amount to the truncated cauchy pdf. If FALSE, proposals for beta are drawn on uniformly around beta, so from bt_can = runif(1, bt_cur - bwb, bt_cur + bwb).

groupMeanComparisons — Logical indicating whether mean comparisons in the form of Bayes Factors and posterior model probabilities should be computed.

skipDichSplit — Logical indicating whether to treat categorical predictor specially. Usually, skipDichSplit = TRUE should be used. This removes the arbitrary dependence on the labeling of categorical predictors and ensures that each group has a regression line of the same shape. If skipDichSplit = FALSE, the model will be the same as lm.circular from the package circular in that no separate treatment for categorical variables is performed.

centerOnly — Logical; If TRUE, the continuous predictors are centered only, not standardized. If FALSE, the continuous predictors are standardized.

Details

The model can be passed either as a combination of a formula and a data frame or matrix data, as in lm(), or as an outcome vector th and a matrix of predictors X. If categorical variables are to be included that are not yet given as dummies, formula syntax is recommended as this will automatically take care of dummy creation.

circGLM performs an MCMC sampler that generates a sample from the posterior of the intercept β_0, regression coefficients β, group mean direction differences δ and residual κ.

An attempt is made to split the predictor matrix `X` into continuous and categorical predictors. This is done so that the categorical predictors can be treated differently, which removes the arbitrary dependence on the labeling of categorical predictors and ensures that each group has a regression line of the same shape.

If categorical predictors are passed as factors, formula syntax is recommended, as it will automatically generate dummy variables. If the predictors are passed as a matrix `X`, categorical variables must be entered as dummy (dichotomous) variables.

The main results obtained are estimates and credible intervals for the parameters, posterior samples, and Bayes factors for various standard hypothesis comparisons.

As with all MCMC samplers, convergence must be checked, and tuning parameters `bwb` and `reparametrize` can be tweaked if the sampler converges poorly. The circGLM object that is returned contains proportions accepted which can be used to monitor performance.

Value

A `circGLM` object, which can be further analyzed with its associated `plot.circGLM`, `coef.circGLM` and `print.circGLM` functions.

An object of class `circGLM` contains the following elements (although some elements are not returned if not applicable):

`b0_meandir` The posterior mean direction of β_0, the circular intercept.

`b0_CCI` The circular credible interval of of β_0, the circular intercept.

`kp_mean` The posterior mean of κ, the concentration parameter.

`kp_mode` The posterior mode of κ, the concentration parameter.

`kp_HDI` The `CIsize` highest posterior density interval of κ.

`kp_propacc` The acceptance proportion of the rejection sampler for κ.

`bt_mean` The posterior means of the regression coefficients β.

`bt_CCI` The credible intervals of the regression coefficients β.

`bt_propacc` The acceptance proportions of the Metropolis-Hastings sampler for β.

`dt_meandir` The posterior mean directions of the group difference parameters, δ.

`dt_CCI` The circular credible intervals of the group difference parameters, δ.

`dt_propacc` The acceptance proportions of the Metropolis-Hastings sampler for δ.

`zt_mean` The posterior means of the reparametrized coefficients ζ.

`zt_mdir` The posterior mean directions of the reparametrized coefficients ζ.

`zt_CCI` The credible intervals of the reparametrized coefficients ζ.

`lppd` Ingredient for information criteria; Log posterior predictive density.

`n_par` Ingredient for information criteria; Number of parameters.

`ll_th_estpars` Ingredient for information criteria; Log-likelihood of the dataset at estimated parameter set.

`ll_each_th_curpars` Ingredient for information criteria; Log-likelihood of each data point at each sampled parameter set.

`ll_th_curpars` Ingredient for information criteria; Log-likelihood of the dataset at each sampled parameter set.

`th_hat` An n-vector of predicted angles.

`b0_chain` A Q-vector of sampled circular intercepts.

`kp_chain` A Q-vector of sampled concentration parameters.

`bt_chain` A matrix of sampled circular regression coefficients.

`dt_chain` A matrix of sampled group difference parameters.

`zt_chain` A matrix of sampled reparametrized circular regression coefficients.

`mu_chain` A matrix of sampled group means.

`AIC_Bayes` A version of the AIC where posterior estimates are used to compute the log-likelihood.

`p_DIC` Ingredient for DIC.

`p_DIC_alt` Ingredient for DIC.

`DIC` The DIC.

`DIC_alt` The alternative formulation of the DIC as given in Bayesian Data Analysis, Gelman et al. (2003).

`p_WAIC1` Ingredient for WAIC1.

`p_WAIC2` Ingredient for WAIC2.

`WAIC1` The first formulation of the WAIC as given in Bayesian Data Analysis, Gelman et al. (2003).

`WAIC2` The second formulation of the WAIC as given in Bayesian Data Analysis, Gelman et al. (2003).

`DeltaIneqBayesFactors` A matrix of inequality Bayes factors for group difference parameters.

`BetaIneqBayesFactors` A matrix of inequality Bayes factors for regression parameters.

`BetaSDDBayesFactors` A matrix of equality Bayes factors (Savage-Dickey Density ratio) for group difference parameters.

`MuIneqBayesFactors` A matrix of inequality Bayes factors for group mean parameters.

`MuSDDBayesFactors` A matrix of equality Bayes factors (Savage-Dickey Density ratio) for group mean parameters.

`SavedIts` Number of iterations returned, without thinned iterations and burn-in.

`TotalIts` Number of iterations performed, including thinning and burn-in.

`TimeTaken` Seconds taken for analysis.

`BetaBayesFactors` Matrix of Bayes factors for regression parameters.

`MuBayesFactors` Matrix of Bayes factors for mean parameters.

`all_chains` A matrix with all sampled values of all parameters.

`Call` The matched call.

`thin` Thinning factor used.

`burnin` Burn-in used.

`data_th` The original dataset.

`data_X` Matrix of used continuous predictors.

`data_d` Matrix of used categorical predictors.

`data_stX` Matrix of used standardized categorical predictors.

`r` Used parameter of the link function.

See Also

print.circGLM, plot.circGLM, coef.circGLM, BF.circGLM, residuals.circGLM, predict.circGLM, predict_function.circGLM, mcmc_summary.circGLM, IC_compare.circGLM.

Examples

```
dat <- generateCircGLMData()
m   <- circGLM(th ~ ., dat)
print(m)
print(m, type = "all")
plot(m, type = "tracestack")
```

---

circglmbayes *circglmbayes: A package for the Bayesian circular GLM.*

Description

This package contains functions to perform a Bayesian circular GLM, which allows regressing a circular outcome on linear and categorical predictors. The model used in this package is similar to the model used by `lm.circular` form the package `circular`. Differences are that the model used by this package treats categorical variables specially. In addition, several hypothesis testing options are provided.

Details

Estimation and uncertainty intervals are all performed in a Bayesian manner through MCMC. Bayesian hypothesis tests are provided through the Bayes factor.

Functions

The main function of the package is circGLM, which runs an MCMC sampler in C++ through Rcpp. This sampler returns an S3 object of type `circGLM`, which can be further analyzed through associated plot.circGLM and print.circGLM functions.

---

circSD *Compute the Circular Standard Deviation*

Description

Returns the circular standard deviation of a vector of circular data which is defined as the square root of minus 2 times the log of the mean resultant length.

Usage

```
circSD(x)
```

Arguments

x	A vector of angles.

Value

A numeric, the circular standard deviation.

---

coef.circGLM *Extract circGLM Coefficients*

Description

Create a table of coefficient results from a circGLM object.

Usage

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

Arguments

object	A circGLM object.
...	Further arguments passed to or from other methods.

Value

A table of coefficients with their corresponding lower and upper bounds.

Examples

```
coef(circGLM(th = rvmc(10, 0, 1)))
```

---

essbhv *Basic Human Values data*

Description

Data from Dutch respondents of the European Social Survey (ESS) on the Basic Human Values scale (Schwartz, 2007).

Usage

```
data(essbhv)
```

Format

A data frame with 1690 rows and 10 variables:

id ESS id number

happy Happiness from 0 (extremely unhappy) to 10 (extremely happy).

rlgdgr Self-reported level of how religious one is from 0 (not at all) to 10 (very religious).

agea Age in years.

edlvenl Highest level of education completed on an 18-point scale.

theta Angle on the Basic Human Values scale in radians from -pi to pi.

thetapos theta plus pi.

thetagrades thetapos converted to degrees.

Details

This dataset includes a circular outcome extracted from the Basic Human Values scale. Note that the extraction of a single circular value as the most salient human value is somewhat of an oversimplification from a theoretical perspective. It is given here because it is nevertheless meaningful as well as useful for illustration purposes.

In addition to the circular outcome, some covariates are included. For further details on the variables included, see the ESS documentation.

Source

ESS Data Portal

References

ESS Round 7: European Social Survey Round 7 Data (2014). Data file edition 2.1. NSD - Norwegian Centre for Research Data, Norway – Data Archive and distributor of ESS data for ESS ERIC.

Schwartz (2007) Basic human values: theory, methods, and application

---

`estimateDensityBySpline` *Estimate the density value from a sample by a spline interpolation of the kernel density*

Description

This function estimates the density at `x0` by first taking a kernel density estimate of a sample from the probability density `x`, and then interpolating it by a spline.

Usage

```
estimateDensityBySpline(x, x0 = 0, npow = 15, rangeExtend = 1/4)
```

Arguments

x	A (large) sample of from the probability density function of interest, such as a posterior.
x0	The value at which to evaluate the density.
npow	The precision used with the density function.
rangeExtend	The number of standard deviations past the range of x to start the density estimate.

Value

Numeric; a scalar of the estimated probability density at x0.

Examples

```
# Compare the estimate from this function with the analytic result.
estimateDensityBySpline(rnorm(1000), 0.1)
dnorm(.1)
```

---

fixResultNames *Fix names for circGLM vector output*

Description

A function to change the names produced in the Rcpp code to more human readable forms.

Usage

```
fixResultNames(nms)
```

Arguments

nms	The original names.

Details

This is only done if the circGLM function is used with output = "vector".

Value

A character vector of the same length as nms.

`generateCircGLMData` *Generate data that follows the circular GLM model*

Description

This function samples data according to the circular GLM model. A set of true values for the parameters can be entered, and a dataset is returned that is drawn from the corresponding model. The link function can also be selected.

Usage

```
generateCircGLMData(n = 30, residkappa = 5, nconpred = 2, ncatpred = 2,
  truebeta0 = pi/2, truebeta = rep(0.25, nconpred), truedelta = rep(1,
  ncatpred), linkfun = function(x) 2 * atan(x))
```

Arguments

`n`	Integer; the sample size to be generated.
`residkappa`	A non-negative numeric; the residual concentration parameter. This is the κ of the von Mises distribution that the residuals follow.
`nconpred`	Integer; The number of continuous (linear) predictors to be generated.
`ncatpred`	Integer; The number of categorical predictors to be generated.
`truebeta0`	An angle in radians representing `beta_0`, which functions as the intercept.
`truebeta`	A numeric vector containing the values for the regression coefficients of the continuous predictors.
`truedelta`	A numeric vector containing angles in radians that represent the group differences for each of the categorical predictors.
`linkfun`	Function; The link function to use. The default is the canonical arctangent link.

Details

This function can also be used as a wrapper for sampling von Mises data, if `nconpred = 0`, `ncatpred = 0`. Then, `beta_0` is the mean of the von Mises distribution and `residkappa` is the concentration parameter κ.

In order to make this function more useful in simulations, the true parameters are also added to the data set that is returned as attributes.

Value

A numeric matrix containing a dataset sampled according to the circular GLM model. The first column `th` represents the circular outcome in radians. The following columns represent the linear predictors and are named `l1`, `l2`, ... The following columns represent the categorical predictors and are named `c1`, `c2`, ... The matrix also has attributes containing the true values of the parameters, the used link function, and a proportion `u` showing the proportion of the data that is on the semicircle closest to `beta_0`.

Examples

```
# Von Mises data with mean 2, kappa 3.
generateCircGLMData(truebeta0 = 2, residkappa = 3,
                    nconpred = 0, ncatpred = 0)

# circGLM data
generateCircGLMData(n = 20, nconpred = 4, truebeta = c(0, 0.4, 0.2, 0.05))
```

`getPMP` *Obtain posterior model probabilities*

Description

Compute posterior model probabilities from odds `x` and a prior odds.

Usage

```
getPMP(x, prior_odds = 1)
```

Arguments

`x`	A vector of odds for which to obtain the posterior model probabilities.
`prior_odds`	The prior odds.

Value

A matrix with two columns, giving the relative probabilities of the first hypothesis versus the second hypothesis.

Examples

```
getPMP(3)
```

IC_compare.circGLM *Compare the information criteria of several circGLM models.*

Description

Compare the information criteria of several circGLM models.

Usage

```
IC_compare.circGLM(..., ICs = c("n_par", "lppd", "AIC_Bayes", "DIC",
  "DIC_alt", "WAIC1", "WAIC2", "p_DIC", "p_DIC_alt", "p_WAIC1", "p_WAIC2"))
```

Arguments

...	The circGLM objects to be compared.
ICs	A character vector of ICs to display.

Value

A matrix with a column of information criteria for each model.

Examples

```
Xcat <- c(rep(0, 5), rep(1, 5))
th <- rvmc(10, 0, 4) + Xcat

# Compare a model that includes group differences with a model that does not.
IC_compare.circGLM(circGLM(th = th), circGLM(th = th, X = Xcat))
```

is.dichotomous *Check if a predictor is dichotomous*

Description

Check if a predictor is dichotomous

Usage

```
is.dichotomous(x)
```

Arguments

x	A character or numerical vector to be tested.

Value

A logical, TRUE if the x has dummy coding (0, 1), FALSE otherwise.

`mcmc_summary.circGLM` *Obtain different central tendencies and CIs from a circGLM object*

Description

Computes the mean (arithmetic or mean direction), median, and mode estimate for the MCMC chains of a `circGLM` object, as well as a credible interval.

Usage

```
mcmc_summary.circGLM(m, modebw = 0.1, ciperc = 0.95)
```

Arguments

`m`	A `circGLM` object.
`modebw`	Numeric between 0 and 1. The modes are estimated by taking the midpoint of a highest density interval. Specifically, the mode is the midpoint of the interval that contains `modebw` of the density of the posterior. Reasonable values are roughly between .005 and .2, although lower values may be reasonable if the number of iterations, Q, is large.
`ciperc`	The confidence interval percentage.

Details

The summary statistics computed have to be computed differently for linear and circular variables.

Value

A matrix with the parameters as rows, and on the columns central tendencies and appropriate credible intervals (circular quantiles and Highest Density Intervals).

Examples

```
dat <- generateCircGLMData()
m   <- circGLM(th ~ ., dat)
mcmc_summary.circGLM(m)
```

medianDirection *Compute the median direction*

Description

This function computes the median direction, which is defined as the middle observation of the shortest arc containing all observations.

Usage

```
medianDirection(th, fastMethod = TRUE)
```

Arguments

th	A vector of angles in radians.
fastMethod	Logical; If TRUE, the data is rotated so that the mean is pi and linear methods are applied. If FALSE, the arcs between each set of data points must be computed, which is much slower. For data that is very strongly spread out, the fast method might not give the correct value.

Value

An angle in radians, the median direction.

Examples

```
medianDirection(rvmc(30, 0, 2))
```

modalDirection *Estimate the modal direction*

Description

Estimates the mode as the midpoint of the highest density interval.

Usage

```
modalDirection(th, modebw = 0.1)
```

Arguments

th	A vector of angles in radians.
modebw	Numeric between 0 and 1. The modes are estimated by taking the midpoint of a highest density interval. Specifically, the mode is the midpoint of the interval that contains modebw of the values of th. Reasonable values are roughly between .005 and .2, although lower values may be reasonable there are a lot of observations in th.

Details

The highest density interval is computed as the shortest interval containing `modebw` of the values in `th`. For circular data however, this definition is not useful, and we should instead look for the shortest arc that contains `modebw` of the data. This is done by rotating the data such that the mean direction is `pi`, and then applying the usual linear methods.

Value

An angle in radians.

Examples

```
modalDirection(rvmc(30, 0, 2))
```

`plot.circGLM` *Plot circGLM object*

Description

General plot function for `circGLM` objects, which dispatches the chosen type of plotting to the corresponding function.

Usage

```
## S3 method for class 'circGLM'
plot(x, type = "trace", ...)
```

Arguments

`x`	A `circGLM` object to be plotted.
`type`	Character string giving the type of plotting. The options are `"trace"`, `"tracestack"`, `"predict"`, `"meancompare"` and `"meanboxplot"`.
`...`	Additional arguments to be passed to subsequent plot functions.

See Also

`plot_trace.circGLM`, `plot_tracestack.circGLM`, `plot_predict.circGLM`, `plot_meancompare.circGLM` and `plot_meanboxplot.circGLM`.

Examples

```
plot(circGLM(th = rvmc(10, 1, 1)))

dat <- generateCircGLMData(n = 100, nconpred = 1, ncatpred = 1)
m   <- circGLM(th ~ ., dat)

# Traceplot by default
plot(m)

# Traceplot stack
plot(m, type = "tracestack")

# Prediction plot
plot(m, type = "predict")

# Mean comparisons
plot(m, type = "meancompare")
plot(m, type = "meanboxplot")
```

`plot_meanboxplot.circGLM` *Plot mean comparison boxplot from circGLM objects*

Description

If the main predictors of interest for the circGLM are categorical, it can be insightful to plot the posteriors of the group means side-by-side, which this function does. This is particularly useful for ANOVA or ANCOVA type designs.

Usage

```
plot_meanboxplot.circGLM(m, xlab = "Mean direction")
```

Arguments

`m`	A `circGLM` object.
`xlab`	The label of the x-axis.

Details

If there are linear predictors in the model as well, the posteriors displayed will correspond to the intercept parameter for each group.

Some caution is needed, as a regular linear boxplot is printed, which may not always be meaningful for a circular variable.

See Also

plot_trace.circGLM, plot_tracestack.circGLM, plot_predict.circGLM, plot_meancompare.circGLM, plot.circGLM.

Examples

```
dat <- generateCircGLMData(nconpred = 0)
m   <- circGLM(th ~ ., dat)
plot_meancompare.circGLM(m)
```

---

plot_meancompare.circGLM

Plot mean comparisons for a circGLM object

Description

If the main predictors of interest for the circGLM are categorical, it can be insightful to plot the posteriors of the group means side-by-side, which this function does. This is particularly useful for ANOVA or ANCOVA type designs.

Usage

```
plot_meancompare.circGLM(m, alpha = 0.7, xlab = "Mean direction")
```

Arguments

m	A circGLM object.
alpha	The transparency (alpha) of the plotted densities.
xlab	The label of the x-axis.

Details

If there are linear predictors in the model as well, the posteriors displayed will correspond to the intercept parameter for each group.

See Also

plot_trace.circGLM, plot_tracestack.circGLM, plot_predict.circGLM, plot_meanboxplot.circGLM, plot.circGLM.

Examples

```
dat <- generateCircGLMData(nconpred = 0)
m   <- circGLM(th ~ ., dat)
plot_meancompare.circGLM(m)
```

plot_predict.circGLM *Create a prediction plot from a circGLM object*

Description

Plot the predictions made by a circGLM analysis.

Usage

```
plot_predict.circGLM(m, x, d, th, linkfun = function(x) m$r * atan(x),
  xlab = NA, ylab = expression(theta), colorPalette = c("#E69F00",
  "#56B4E9"))
```

Arguments

m	A circGLM object.
x	Optional; Either a numeric vector with a continuous predictor or string naming the desired variable to plot on the x-axis. If missing, we just use the first continuous predictor in the circGLM object.
d	Optional; Either a numeric vector with a categorical predictor or string naming the desired variable to plot on the x-axis. If missing, we just use the first categorical predictor in the circGLM object.
th	Optional; Can be a new numeric vector containing outcome angles corresponding to predictors x and potentially d.
linkfun	The link function to be used. Should be the same as was used for the creation of the circGLM object.
xlab	A character string with the x-label.
ylab	A character string with the y-label.
colorPalette	The colors to use in plotting, max 2.

Details

Creates a ggplot showing a prediction plot showing linear predictor against the circular outcome, with an optional grouping variable. One or more regression lines show the predicted values for different values of the linear and categorical predictors.

Predictors x and d and outcome th can be provided as numeric vectors of the same length as the outcome in the circGLM object m. This allows plotting the regression line from an earlier dataset on a new dataset.

Alternatively, x and d can be strings containing names of corresponding predictors in the original model. In that case, th should not be provided.

The function makes an effort to find predictors to plot if none are given, where it will simply take the first predictor in the dataset. If a plot without grouping is required, d can be set to NA.

Value

A ggplot, to which further ggplot elements can be added.

See Also

plot_trace.circGLM, plot_tracestack.circGLM, plot_meancompare.circGLM, plot_meanboxplot.circGLM, plot.circGLM.

Examples

```
dat <- generateCircGLMData()
m   <- circGLM(th ~ ., dat)
plot(m, type = "predict")
```

---

plot_trace.circGLM *Make traceplots for circGLM*

Description

Plot traceplots from a circGLM object. This plotting method uses the standard coda traceplots.

Usage

```
plot_trace.circGLM(m, params, ...)
```

Arguments

m	A circGLM object.
params	An optional character vector containing the parameter chains to display. If left empty, all are plotted.
...	Additional parameters passed to plot.mcmc from the coda package.

See Also

plot_tracestack.circGLM, plot_predict.circGLM, plot_meancompare.circGLM, plot_meanboxplot.circGLM, plot.circGLM.

Examples

```
plot_trace.circGLM(circGLM(th = rvmc(10, 1, 1)))

dat <- generateCircGLMData()
plot(circGLM(th ~., dat), type = "trace")
```

plot_tracestack.circGLM *Plot a stack of traceplots for a circGLM object*

Description

An alternative option to plot traceplots from `circGLM` objects.

Usage

```
plot_tracestack.circGLM(m, coef = "Beta", labelFormat = "default",
  ggTheme = ggplot2::theme_bw(), res = 10000, burnThinLabel = TRUE)
```

Arguments

`m`	A `circGLM` object.
`coef`	A character string, either "Beta" or "Zeta", determining whether the continuous regression predictors are shown in reparametrized form or not.
`labelFormat`	A character vector, either `"default"`, `"numbered"` or `"latex"`. By default, we find the names of the variables in the circGLM object. If `"numbered"`, the parameter names are numbered. The "latex" labels are useful if `knitr` is used with a Tikz device.
`ggTheme`	A ggplot theme object to use. The relevant theme function should be evaluated.
`res`	The maximum number iterations to print. If `res` is larger than the number of iterations in the `circGLM` object, a subset of size `res` is selected, and it is attempted to equally space the selected iterations from the full set. This is useful if there is a very large posterior sample due to having very little thinning.
`burnThinLabel`	Logical; if TRUE, the x-label will reflect the fact that a burn-in and a thinning factor were used. If FALSE, the x-labels will run from 1 to Q.

Value

A ggplot2 plot.

See Also

`plot_trace.circGLM`, `plot_predict.circGLM`, `plot_meancompare.circGLM`, `plot_meanboxplot.circGLM`, `plot.circGLM`.

Examples

```
plot(circGLM(th = rvmc(100, 0, 1)), type = "tracestack")

dat <- generateCircGLMData()
plot(circGLM(th ~ ., dat), type = "tracestack")
```

predict.circGLM *Obtain predictions for the circGLM model*

Description

Obtain predictions from the original dataset, or the predictions from the fitted model on a new dataset newdata.

Usage

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

Arguments

object	A circGLM object.
newdata	A data frame with predictors. The predictors must be the same as used in the circGLM object and must have the same column names.
...	Further arguments passed to or from other methods.

Value

A numeric vector with predictions.

Examples

```
dat <- generateCircGLMData()
m   <- circGLM(th ~ ., dat)

# Predictions for the original outcome angles.
predict(m)

# Predictions for new data
dat2  <- generateCircGLMData()
predict(m, newdata = dat2)
```

predict_function.circGLM *Obtain a prediction function from a circGLM object*

Description

This functions creates and returns a new prediction function that takes in new data, and returns their predicted values. The prediction function is based on the posterior estimates.

Usage

```
predict_function.circGLM(object, linkfun = function(x) atanLF(x, 2))
```

Arguments

object	A circGLM object.
linkfun	A link function to use in the analysis. Should be the same as the link function.

Value

A function that takes newdata as an argument, which must be a data frame with predictors. The predictors must be the same as used in the circGLM object and must have the same column names.

Examples

```
dat <- generateCircGLMData()
m   <- circGLM(th ~ ., dat)
predfun <- predict_function.circGLM(m)
newd <- generateCircGLMData()

# Predicted values of the new data.
predfun(newd)
```

---

print.circGLM *Print circGLM Object*

Description

General print function for circGLM objects, which dispatches the chosen type of printing to the corresponding function.

Usage

```
## S3 method for class 'circGLM'
print(x, type = "text", ...)
```

Arguments

x	A circGLM object to be printed.
type	Character string giving the type of printing, such as "text", "mcmc", "all", "coef".
...	Additional arguments to be passed to print functions.

See Also

print_text.circGLM, print_mcmc.circGLM, print_all.circGLM, print_coef.circGLM.

Examples

```
print(circGLM(th = rvmc(10, 1, 1)))

dat <- generateCircGLMData()
cglmmod <- circGLM(th ~ ., dat)

print(cglmmod)

print(cglmmod, type = "mcmc")

print(cglmmod, type = "all")

print(cglmmod, type = "coef")
```

---

print_all.circGLM *Print all results from a circGLM object*

Description

This function prints the full list of results from a `circGLM` object. The function extracts all the scalar results and displays these together, then prints all further list elements. The full chains are not printed.

Usage

```
print_all.circGLM(m, digits = 3)
```

Arguments

`m`	A `circGLM` object.
`digits`	Number of digits to display.

See Also

`print_text.circGLM`, `print_mcmc.circGLM`, `print_coef.circGLM`, `print.circGLM`.

Examples

```
print(circGLM(th = rvmc(10, 1, 1)), type = "all")

dat <- generateCircGLMData()
cglmmod <- circGLM(th ~ ., dat)
print(cglmmod, type = "all")
```

print_coef.circGLM *Print circGLM coefficients*

Description

Print circGLM coefficients

Usage

```
print_coef.circGLM(m, digits = 3)
```

Arguments

m	A circGLM object.
digits	Number of digits to display.

See Also

print_text.circGLM, print_mcmc.circGLM, print_all.circGLM, print.circGLM.

Examples

```
print(circGLM(th = rvmc(10, 0, 1)), type = "coef")

dat <- generateCircGLMData()
cglmmod <- circGLM(th = dat[, 1], X = dat[, -1])
print(cglmmod, type = "coef")
```

print_mcmc.circGLM *Print the mcmc results from a circGLM object*

Description

This prints a number of diagnostics about the results of a circGLM objects through summary.mcmc from the coda package. In particular, the standard errors may be of interest.

Usage

```
print_mcmc.circGLM(m, ...)
```

Arguments

m	A circGLM object.
...	Additional arguments to be passed to coda printing functions.

Details

Note that the standard error and convergence diagnostics computed by coda are not necessarily trustworthy.

See Also

print_text.circGLM, print_all.circGLM, print_coef.circGLM, print.circGLM.

Examples

```
print(circGLM(th = rvmc(10, 1, 1)), type = "mcmc", digits = 3)

dat <- generateCircGLMData()
cglmmod <- circGLM(th = dat[, 1], X = dat[, -1])
print(cglmmod, type = "mcmc")
```

---

print_text.circGLM *Print the main results from a* circGLM *object.*

Description

Print the main results from a circGLM object.

Usage

```
print_text.circGLM(m, digits = 3)
```

Arguments

m	A circGLM object.
digits	Number of digits to display.

See Also

print_mcmc.circGLM, print_all.circGLM, print_coef.circGLM, print.circGLM.

Examples

```
print(circGLM(th = rvmc(10, 1, 1)), type = "text")

dat <- generateCircGLMData()
cglmmod <- circGLM(th = dat[, 1], X = dat[, -1])
print(cglmmod, type = "text")
```

residuals.circGLM *Obtain residuals from a circGLM object*

Description

Computes the residuals either by taking the arc distance or the cosine distance between the predictions and the observed outcomes.

Usage

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

Arguments

object	A circGLM object.
type	Either "arc" or "cosine", the type of distance to take.
...	Further arguments passed to or from other methods.

Value

A numeric vector of residuals. If type is "arc", these are angles in radians. If type is "cosine", these are numeric values between 0 and 2.

Examples

```
m <- circGLM(th = rvmc(10, 0, 1))
residuals(m)

# Cosine residuals
residuals(m, type = "cosine")
```

rvmc *Generate a random variate from the von Mises distribution*

Description

This function generates a set of data from the von Mises distribution. If kappa is very small, return a circular uniform draw, as otherwise the algorithm will fail.

Usage

```
rvmc(n, mu, kp)
```

Arguments

n	The number of random variates required.
mu	The required mean direction, mu.
kp	The required concentration, kappa.

Value

A vector of length n containing VM random variates.

---

sampleKappa *Sample a value from the Bessel exponential distribution*

Description

This is an implementation of the algorithm of Forbes and Mardia (2015) to sample from the Bessel exponential distribution, which is the conditional distribution of the concentration parameter of a von Mises distribution given the mean mu. The distribution is proportional to $exp(-\eta g \kappa)/I_0(\kappa)^\eta$. Note that beta_0 in Forbes and Mardia (2015) is renamed g here.

Usage

```
sampleKappa(etag, eta)
```

Arguments

etag	Numeric; This is eta * g, which should -R*cos(mu-theta_bar), where R is the posterior mean resultant length, and theta_bar is the posterior mean, while mu is the current value of the mean.
eta	Integer; This is the posterior sample size, which is n + c where c is the number of observations contained in the conjugate prior. For uninformative, c = 0 and eta = n.

Value

A sampled value kappa from the Bessel exponential distribution.

Index