Package 'ggmcmc'

January 4, 2016

Title Tools for Analyzing MCMC Simulations from Bayesian Inference

Description Tools for assessing and diagnosing convergence of Markov Chain Monte Carlo simulations, as well as for graphically display results from full MCMC analysis. The package also facilitates the graphical interpretation of models by providing flexible functions to plot the results against observed variables.

Version 0.7.3

Maintainer Xavier Fernández i Marín <xavier.fim@gmail.com>

Author Xavier Fernández i Marín <xavier.fim@gmail.com>

Depends dplyr, tidyr (>= 0.3.1), ggplot2

Imports GGally (>= 0.5.0)

Suggests coda

License GPL-2

URL http://xavier-fim.net/packages/ggmcmc, https://github.com/xfim/ggmcmc

BugReports https://github.com/xfim/ggmcmc/issues

Encoding UTF-8

Collate 'ggmcmc.R' 'ggs.R' 'ggs_autocorrelation.R' 'ggs_compare_partial.R' 'ggs_crosscorrelation.R' 'ggs_density.R' 'ggs_histogram.R' 'ggs_running.R' 'ggs_traceplot.R' 'ggs_pairs.R' 'data.R' 'help.R' 'functions.R' 'ggs_Rhat.R' 'ggs_geweke.R' 'ggs_caterpillar.R' 'ggs_separation.R' 'globals.R' 'ggs_ppmean.R' 'ggs_ppsd.R' 'ggs_rocplot.R'

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2016-01-04 12:47:41

R topics documented:

---

ac *Calculate the autocorrelation of a single chain, for a specified amount of lags*

Description

Calculate the autocorrelation of a single chain, for a specified amount of lags.

Usage

```
ac(x, nLags)
```

Arguments

x	Vector with a chain of simulated values.
nLags	Numerical value with the maximum number of lags to take into account.

Details

Internal function used by ggs_autocorrelation.

Value

A matrix with the autocorrelations of every chain.

Examples

```
# Calculate the autocorrelation of a simple vector
ac(cumsum(rnorm(10))/10, nLags=4)
```

---

calc_bin *Calculate binwidths by parameter, based on the total number of bins.*

Description

Compute the minimal elements to recreate a histogram manually by defining the total number of bins.

Usage

```
calc_bin(x, bins = bins)
```

Arguments

x	any vector or variable
bins	the number of requested bins

Details

Internal function to compute the minimal elements to recreate a histogram manually by defining the total number of bins, used by ggs_histogram ggs_ppmean and ggs_ppsd.

Value

A data frame with the x location, the width of the bars and the number of observations at each x location.

ci *Calculate Credible Intervals (wide and narrow).*

Description

Generate a data frame with the limits of two credible intervals. Function used by ggs_caterpillar. "low" and "high" refer to the wide interval, whereas "Low" and "High" refer to the narrow interval. "median" is self-explanatory and is used to draw a dot in caterpillar plots. The data frame generated is of wide format, suitable for ggplot2::geom_segment().

Usage

```
ci(D, thick_ci = c(0.05, 0.95), thin_ci = c(0.025, 0.975))
```

Arguments

D	Data frame whith the simulations.
thick_ci	Vector of length 2 with the quantiles of the thick band for the credible interval
thin_ci	Vector of length 2 with the quantiles of the thin band for the credible interval

Value

A data frame tbl with the Parameter names and 5 variables with the limits of the credibal intervals (thin and thick), ready to be used to produce caterpillar plots.

Examples

```
data(linear)
ci(ggs(s))
```

get_family *Subset a ggs object to get only the parameters with a given regular expression.*

Description

Internal function used by the graphical functions to get only some of the parameters that follow a given regular expression.

Usage

```
get_family(D, family = NA)
```

Arguments

D	Data frame with the data arranged and ready to be used by the rest of the ggmcmc functions. The dataframe has four columns, namely: Iteration, Parameter, value and Chain, and six attributes: nChains, nParameters, nIterations, nBurnin, nThin and description.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).

Value

D Data frame that is a subset of the given D dataset.

---

ggmcmc *Wrapper function that creates a single pdf file with all plots that ggmcmc can produce.*

Description

`ggmcmc()` is simply a wrapper function that generates a pdf file with all the potential plots that the package can produce.

ggmcmc is a tool for assessing and diagnosing convergence of Markov Chain Monte Carlo simulations, as well as for graphically display results from full MCMC analysis. The package also facilitates the graphical interpretation of models by providing flexible functions to plot the results against observed variables.

Usage

```
ggmcmc(D, file = "ggmcmc-output.pdf", family = NA, plot = NULL,
  param_page = 5, width = 7, height = 10, simplify_traceplot = NULL,
  ...)
```

Arguments

D	Data frame whith the simulations, previously arranged using ggs
file	Character vector with the name of the file to create. Defaults to "ggmcmc-output.pdf". When NULL, no pdf device is opened or closed. This allows the user to work with an opened pdf (or other) device.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).

plot	character vector containing the names of the desired plots. By default (NULL), ggmcmc() plots ggs_histogram(), ggs_density(), ggs_traceplot(), ggs_running(), ggs_compare_partial(), ggs_autocorrelation(), ggs_crosscorrelation(), ggs_Rhat(), ggs_geweke() and ggs_caterpillar().
param_page	Numerical, number of parameters to plot for each page. Defaults to 5.
width	Width of the pdf display, in inches. Defaults to 7.
height	Height of the pdf display, in inches. Defaults to 10.
simplify_traceplot	Numerical. A percentage of iterations to keep in the time series. It is an option intended only for the purpose of saving time and resources when doing traceplots. It is not a thin operation, because it is not regular. It must be used with care.
...	Other options passed to the pdf device.

Details

Notice that caterpillar plots are only created when there are multiple parameters within the same family. A family of parameters is considered to be all parameters that have the same name (usually the same greek letter) but different number within square brackets (such as alpha[1], alpha[2], ...).

References

http://xavier-fim.net/packages/ggmcmc.

Examples

```
data(linear)
ggmcmc(ggs(s))  # Directly from a coda object
```

---

ggs *Import MCMC samples into a ggs object than can be used by all ggs_* graphical functions.*

Description

This function manages MCMC samples from different sources (JAGS, MCMCpack, STAN -both via rstan and via csv files-) and converts them into a data frame tbl. The resulting data frame has four columns (Iteration, Chain, Parameter, value) and six attributes (nChains, nParameters, nIterations, nBurnin, nThin and description). The ggs object returned is then used as the input of the ggs_* functions to actually plot the different convergence diagnostics.

Usage

```
ggs(S, family = NA, description = NA, burnin = TRUE, par_labels = NA,
  inc_warmup = FALSE, stan_include_auxiliar = FALSE)
```

Arguments

S	Either a mcmc.list object with samples from JAGS, a mcmc object with samples from MCMCpack, a stanfit object with samples from rstan, or a list with the filenames of csv files generated by stan outside rstan (where the order of the files is assumed to be the order of the chains). ggmcmc guesses what is the original object and tries to import it accordingly. rstan is not expected to be in CRAN soon, and so coda::mcmc is used to extract stan samples instead of the more canonical rstan::extract.
family	Name of the family of parameters to process, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
description	Character vector giving a short descriptive text that identifies the model.
burnin	Logical or numerical value. When logical and TRUE (the default), the number of samples in the burnin period will be taken into account, if it can be guessed by the extracting process. Otherwise, iterations will start counting from 1. If a numerical vector is given, the user then supplies the length of the burnin period.
par_labels	data frame with two colums. One named "Parameter" with the same names of the parameters of the model. Another named "Label" with the label of the parameter. When missing, the names passed to the model are used for representation. When there is no correspondence between a Parameter and a Label, the original name of the parameter is used. The order of the levels of the original Parameter does not change.
inc_warmup	Logical. When dealing with stanfit objects from rstan, logical value whether the warmup samples are included. Defaults to FALSE.
stan_include_auxiliar	Logical value to include "lp__" parameter in rstan, and "lp__", "treedepth__" and "stepsize__" in stan running without rstan. Defaults to FALSE.

Value

D A data frame tbl with the data arranged and ready to be used by the rest of the ggmcmc functions. The data frame has four columns, namely: Iteration, Chain, Parameter and value, and six attributes: nChains, nParameters, nIterations, nBurnin, nThin and description. A data frame tbl is a wrapper to a local data frame, behaves like a data frame and its advantage is related to printing, which is compact. For more details, see tbl_df() in package dplyr.

Examples

```
# Assign 'D' to be a data frame suitable for \code{ggmcmc} functions from
# a coda object called S
data(linear)
S <- ggs(s)        # s is a coda object

# Get samples from 'beta' parameters only
S <- ggs(s, family = "beta")
```

ggs_autocorrelation *Plot an autocorrelation matrix*

Description

Plot an autocorrelation matrix.

Usage

```
ggs_autocorrelation(D, family = NA, nLags = 50)
```

Arguments

D	Data frame whith the simulations.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
nLags	Integer indicating the number of lags of the autocorrelation plot.

Value

A ggplot object.

Examples

```
data(linear)
ggs_autocorrelation(ggs(s))
```

ggs_caterpillar *Caterpillar plot with thick and thin CI*

Description

Caterpillar plots are plotted combining all chains for each parameter.

Usage

```
ggs_caterpillar(D, family = NA, X = NA, thick_ci = c(0.05, 0.95),
  thin_ci = c(0.025, 0.975), line = NA, horizontal = TRUE,
  model_labels = NULL)
```

Arguments

D	Data frame whith the simulations or list of data frame with simulations. If a list of data frames with simulations is passed, the names of the models are the names of the objects in the list.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
X	data frame with two columns, Parameter and the value for the x location. Parameter must be a character vector with the same names that the parameters in the D object.
thick_ci	Vector of length 2 with the quantiles of the thick band for the credible interval
thin_ci	Vector of length 2 with the quantiles of the thin band for the credible interval
line	Numerical value indicating a concrete position, usually used to mark where zero is. By default do not plot any line.
horizontal	Logical. When TRUE (the default), the plot has horizontal lines. When FALSE, the plot is reversed to show vertical lines. Horizontal lines are more appropriate for categorical caterpillar plots, because the x-axis is the only dimension that matters. But for caterpillar plots against another variable, the vertical position is more appropriate.
model_labels	Vector of strings that matches the number of models in the list. It is only used in case of multiple models and when the list of ggs objects given at D is not named. Otherwise, the names in the list are used.

Value

A ggplot object.

Examples

```
data(linear)
ggs_caterpillar(ggs(s))
ggs_caterpillar(list(A=ggs(s), B=ggs(s))) # silly example duplicating the same model
```

---

ggs_chain *Auxiliary function that extracts information from a single chain.*

Description

Auxiliary function that extracts information from a single chain.

Usage

```
ggs_chain(s)
```

Arguments

s	a single chain to convert into a data frame

Value

D data frame with the chain arranged

`ggs_compare_partial` *Density plots comparing the distribution of the whole chain with only its last part.*

Description

Density plots comparing the distribution of the whole chain with only its last part.

Usage

```
ggs_compare_partial(D, family = NA, partial = 0.1, rug = FALSE)
```

Arguments

D	Data frame whith the simulations
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
partial	Percentage of the chain to compare to. Defaults to the last 10 percent.
rug	Logical indicating whether a rug must be added to the plot. It is FALSE by default, since in large chains it may use lot of resources and it is not central to the plot.

Value

A `ggplot` object.

Examples

```
data(linear)
ggs_compare_partial(ggs(s))
```

ggs_crosscorrelation *Plot the Cross-correlation between-chains*

Description

Plot the Cross-correlation between-chains.

Usage

```
ggs_crosscorrelation(D, family = NA, absolute_scale = TRUE)
```

Arguments

D	Data frame whith the simulations.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
absolute_scale	Logical. When TRUE (the default), the scale of the colour diverges between perfect inverse correlation (-1) to perfect correlation (1), whereas when FALSE, the scale is relative to the minimum and maximum cross-correlations observed.

Value

a ggplot object.

Examples

```
data(linear)
ggs_crosscorrelation(ggs(s))
```

ggs_density *Density plots of the chains*

Description

Density plots with the parameter distribution. For multiple chains, use colours to differentiate the distributions.

Usage

```
ggs_density(D, family = NA, rug = FALSE)
```

Arguments

D	Data frame whith the simulations.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
rug	Logical indicating whether a rug must be added to the plot. It is FALSE by default, since in large chains it may use lot of resources and it is not central to the plot.

Value

A ggplot object.

Examples

```
data(linear)
ggs_density(ggs(s))
```

---

ggs_geweke *Dotplot of the Geweke diagnostic, the standard Z-score*

Description

Dotplot of Geweke diagnostic.

Usage

```
ggs_geweke(D, family = NA, frac1 = 0.1, frac2 = 0.5,
  shadow_limit = TRUE)
```

Arguments

D	data frame whith the simulations.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
frac1	Numeric, proportion of the first part of the chains selected. Defaults to 0.1.
frac2	Numeric, proportion of the last part of the chains selected. Defaults to 0.5.
shadow_limit,	logical. When TRUE (the default), a shadowed area between -2 and +2 is drawn.

Value

A ggplot object.

Examples

```
data(linear)
ggs_geweke(ggs(s))
```

`ggs_histogram` *Histograms of the paramters.*

Description

Plot a histogram of each of the parameters. Histograms are plotted combining all chains for each parameter.

Usage

```
ggs_histogram(D, family = NA, bins = 30)
```

Arguments

`D`	Data frame whith the simulations.
`family`	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
`bins`	integer indicating the total number of bins in which to divide the histogram. Defaults to 30, which is the same as geom_histogram()

Value

A `ggplot` object.

Examples

```
data(linear)
ggs_histogram(ggs(s))
```

ggs_pairs *Create a plot matrix of posterior simulations*

Description

Pairs style plots to evaluate posterior correlations among parameters.

Usage

```
ggs_pairs(D, family = NA, ...)
```

Arguments

D	Data frame with the simulations.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
...	Arguments to be passed to ggpairs, including geom's aes (see examples)

Value

A ggpairs object that creates a plot matrix consisting of univariate density plots on the diagonal, correlation estimates in upper triangular elements, and scatterplots in lower triangular elements.

Examples

```
library(GGally)
data(linear)

# default ggpairs plot
ggs_pairs(ggs(s))

# change alpha transparency of points
ggs_pairs(ggs(s), lower=list(continuous = wrap("points", alpha = 0.2)))

# with too many points, try contours instead
ggs_pairs(ggs(s), lower=list(continuous="density"))

# histograms instead of univariate densities on diagonal
ggs_pairs(ggs(s), diag=list(continuous="barDiag"))

# coloring results according to chains
ggs_pairs(ggs(s), mapping = aes(color = Chain))

# custom points on lower panels, black contours on upper panels
ggs_pairs(ggs(s),
  upper=list(continuous = wrap("density", color = "black")),
  lower=list(continuous = wrap("points", alpha = 0.2, shape = 1)))
```

`ggs_ppmean` *Posterior predictive plot comparing the outcome mean vs the distribution of the predicted posterior means.*

Description

Histogram with the distribution of the predicted posterior means, compared with the mean of the observed outcome.

Usage

```
ggs_ppmean(D, outcome, family = NA, bins = 30)
```

Arguments

`D`	Data frame whith the simulations. Notice that only the posterior outcomes are needed, and so either the ggs() call limits the parameters to the outcomes or the user provides a family of parameters to limit it.
`outcome`	vector (or matrix or array) containing the observed outcome variable. Currently only a vector is supported.
`family`	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
`bins`	integer indicating the total number of bins in which to divide the histogram. Defaults to 30, which is the same as geom_histogram()

Value

A `ggplot` object.

Examples

```
data(linear)
ggs_ppmean(ggs(s.y.rep), outcome=y)
```

`ggs_ppsd` *Posterior predictive plot comparing the outcome standard deviation vs the distribution of the predicted posterior standard deviations.*

Description

Histogram with the distribution of the predicted posterior standard deviations, compared with the standard deviations of the observed outcome.

Usage

```
ggs_ppsd(D, outcome, family = NA, bins = 30)
```

Arguments

D	Data frame whith the simulations. Notice that only the posterior outcomes are needed, and so either the ggs() call limits the parameters to the outcomes or the user provides a family of parameters to limit it.
outcome	vector (or matrix or array) containing the observed outcome variable. Currently only a vector is supported.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
bins	integer indicating the total number of bins in which to divide the histogram. Defaults to 30, which is the same as geom_histogram()

Value

A ggplot object.

Examples

```
data(linear)
ggs_ppsd(ggs(s.y.rep), outcome=y)
```

---

ggs_Rhat *Dotplot of Potential Scale Reduction Factor (Rhat)*

Description

Plot a dotplot of Potential Scale Reduction Factor (Rhat), proposed by Gelman and Rubin (1992). The version from the second edition of Bayesian Data Analysis (Gelman, Carlin, Stein and Rubin) is used.

Usage

```
ggs_Rhat(D, family = NA, scaling = 1.5)
```

Arguments

D	Data frame whith the simulations
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
scaling	Value of the upper limit for the x-axis. By default, it is 1.5, to help contextualization of the convergence. When 0 or NA, the axis are not scaled.

Details

Notice that at least two chains are required.

Value

A ggplot object.

Examples

```
data(linear)
ggs_Rhat(ggs(s))
```

---

ggs_rocplot *Receiver-Operator Characteristic (ROC) plot for models with binary outcomes*

Description

Receiver-Operator Characteristic (ROC) plot for models with binary outcomes

Usage

```
ggs_rocplot(D, outcome, fully_bayesian = FALSE)
```

Arguments

D	Data frame whith the simulations. Notice that only the posterior outcomes are needed, and so either the previous call to ggs() should have limited the family of parameters to pass to the predicted outcomes.
outcome	vector (or matrix or array) containing the observed outcome variable. Currently only a vector is supported.
fully_bayesian	logical, false by default. When not fully Bayesian, it uses the median of the predictions for each observation by iteration. When TRUE the function plots as many ROC curves as iterations. It uses a a lot of CPU and needs more memory. Use it with caution.

Value

A ggplot object

Examples

```
data(binary)
ggs_rocplot(ggs(s.binary, family="mu"), outcome=y.binary)
```

ggs_running *Running means of the chains*

Description

Running means of the chains.

Usage

```
ggs_running(D, family = NA, original_burnin = TRUE, original_thin = TRUE)
```

Arguments

D	Data frame whith the simulations.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).
original_burnin	Logical. When TRUE (the default), start the iteration counter in the x-axis at the end of the burnin period.
original_thin	Logical. When TRUE (the default), take into account the thinning interval in the x-axis.

Value

A ggplot object.

Examples

```
data(linear)
ggs_running(ggs(s))
```

ggs_separation *Separation plot for models with binary response variables*

Description

Separation plot for models with binary response variables

Usage

```
ggs_separation(D, outcome, fully_bayesian = FALSE, minimalist = FALSE)
```

Arguments

D	Data frame whith the simulations. Notice that only the fitted / expected posterior outcomes are needed, and so either the previous call to ggs() should have limited the family of parameters to only pass the fitted / expected values. See the example below.
outcome	vector (or matrix or array) containing the observed outcome variable. Currently only a vector is supported.
fully_bayesian	logical, FALSE by default. Currently not implemented
minimalist	logical, FALSE by default. It returns a minimalistic version of the figure with the bare minimum elements, suitable for being used inline as suggested by Greenhill, Ward and Sacks citing Tufte.

Value

A ggplot object

References

Greenhill, Ward and Sacks (2011): The separation plot: a new visual method for evaluating the fit of binary models. American Journal of Political Science, vol 55, number 4, pg 991-1002.

Examples

```
data(binary)
ggs_separation(ggs(s.binary, family="mu"), outcome=y.binary)
```

---

ggs_traceplot *Traceplot of the chains*

Description

Traceplot with the time series of the chains.

Usage

```
ggs_traceplot(D, family = NA, original_burnin = TRUE,
  original_thin = TRUE, simplify = NULL)
```

Arguments

D	Data frame with the simulations.
family	Name of the family of parameters to plot, as given by a character vector or a regular expression. A family of parameters is considered to be any group of parameters with the same name but different numerical value between square brackets (as beta[1], beta[2], etc).

`original_burnin`
: Logical. When TRUE (the default) start the Iteration counter in the x-axis at the end of the burnin period.

`original_thin`
: Logical. When TRUE (the default) take into account the thinning interval in the x-axis.

`simplify`
: Numerical. A percentage of iterations to keep in the time series. It is an option intended only for the purpose of saving time and resources when doing trace-plots. It is not a thin operation, because it is not regular. It must be used with care.

Value

A `ggplot` object.

Examples

```
data(linear)
ggs_traceplot(ggs(s))
```

---

gl_unq *Generate a factor with unequal number of repetitions.*

Description

Generate a factor with levels of unequal length.

Usage

```
gl_unq(n, k, labels = 1:n)
```

Arguments

`n`	number of levels
`k`	number of repetitions
`labels`	optional vector of labels

Details

Internal function to generate a factor with levels of unequal length, used by `ggs_histogram`.

Value

A factor

radon — *Simulations of the parameters of a hierarchical model using the radon example in Gelman & Hill (2007).*

Description

A list containing the following elements: `counties` a data frame with the county label, ids and radon level; `id.county` a vector identifying counties in the data; `y` the outcome variable; `s.radon` a coda object with simulated values from the posterior distribution of all parameters, with few iterations for each one; `s.radon.yhat` a coda object containing simulated values from the posterior predictive distribution; and `s.radon.short` a coda object with simulated values from the posterior distribution of few parameters, with reasonable chain length. The purpose of the object is only to show the possibilities of the ggmcmc package.

Usage

```
radon
```

Format

A list containing several elements to show the possibilities af ggmcmc.

roc_calc — *Calculate the ROC curve for a set of observed outcomes and predicted probabilities*

Description

Internal function used by `ggs_autocorrelation`.

Usage

```
roc_calc(R)
```

Arguments

`R`	data frame with the 'value' (predicted probability) and the observed 'Outcome'.

Value

A data frame with the Sensitivity and the Specificity.

s — *Simulations of the parameters of a simple linear regression with fake data.*

Description

A coda object containing simulated values from the posterior distribution of the intercept, slope and residual of a linear regression with fake data (y = beta[1] + beta[2] * X + sigma). The purpose of the dataset is only to show the possibilities of the ggmcmc package.

Usage

```
s
```

Format

A coda object containing posterior distributions of the intercept, slope and residual of a linear regression with fake data.

s.binary — *Simulations of the parameters of a simple linear regression with fake data.*

Description

A coda object containing simulated values from the posterior distribution of the intercept and slope of a logistic regression with fake data (y ~ dbern(mu); logit(mu) = theta[1] + theta[2] * X), and the fitted / expected values (mu). The purpose of the dataset is only to show the possibilities of the ggmcmc package.

Usage

```
s.binary
```

Format

A coda object containing posterior distributions of the intercept (theta[1]) and slope (theta[2]) of a logistic regression with fake data, and of the fitted / expected values (mu).

s.y.rep — *Simulations of the posterior predictive distribution of a simple linear regression with fake data.*

Description

A coda object containing simulated values from the posterior predictive distribution of the outcome of a linear regression with fake data (y ~ N(mu, sigma); mu = beta[1] + beta[2] * X; y.rep ~ N(mu, sigma); where y.rep is a replicated outcome, originally missing data). The purpose of the dataset is only to show the possibilities of the ggmcmc package.

Usage

```
s.y.rep
```

Format

A coda object containing posterior distributions of the posterior predictive distribution of a linear regression with fake data.

sde0f — *Spectral Density Estimate at Zero Frequency.*

Description

Compute the Spectral Density Estimate at Zero Frequency for a given chain.

Usage

```
sde0f(x)
```

Arguments

x	A time series

Details

Internal function to compute the Spectral Density Estimate at Zero Frequency for a given chain used by ggs_geweke.

Value

A vector with the spectral density estimate at zero frequency

---

`y` *Values for the observed outcome of a simple linear regression with fake data.*

Description

A numeric vector containing the observed values of the outcome of a linear regression with fake data (y = beta[1] + beta[2] + X + sigma). The purpose of the dataset is only to show the possibilities of the ggmcmc package.

Usage

```
y
```

Format

A numeric vector containing the observed values of the outcome in the linear regression with fake data.

`y.binary` *Values for the observed outcome of a binary logistic regression with fake data.*

Description

A numeric vector containing the observed values (y) of the outcome of a logistic regression with fake data (y ~ dbern(mu); logit(mu) = theta[1] + theta[2] * X). The purpose of the dataset is only to show the possibilities of the ggmcmc package.

Usage

```
y.binary
```

Format

A numeric vector containing the observed values of the outcome in the linear regression with fake data.

Index