Package ‘ggmcmc’

April 2, 2020

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 1.4.1

Depends R (>= 3.5), dplyr (>= 0.5.0), tidyr (>= 0.5.1), ggplot2

Imports GGally (>= 1.1.0)

Suggests coda, rstan (>= 2.10.1), nimble, knitr, rmarkdown, ggthemes,
gridExtra, Cairo, extrafont

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

RoxygenNote 7.1.0

VignetteBuilder knitr

NeedsCompilation no

Author Xavier Fernández i Marín [aut, cre]

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

Repository CRAN

Date/Publication 2020-04-02 11:00:02 UTC
**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ac</td>
<td>3</td>
</tr>
<tr>
<td>binary</td>
<td>3</td>
</tr>
<tr>
<td>calc_bin</td>
<td>4</td>
</tr>
<tr>
<td>ci</td>
<td>5</td>
</tr>
<tr>
<td>custom.sort</td>
<td>5</td>
</tr>
<tr>
<td>get_family</td>
<td>6</td>
</tr>
<tr>
<td>ggmcmc</td>
<td>6</td>
</tr>
<tr>
<td>ggs</td>
<td>8</td>
</tr>
<tr>
<td>ggs_autocorrelation</td>
<td>10</td>
</tr>
<tr>
<td>ggs_caterpillar</td>
<td>11</td>
</tr>
<tr>
<td>ggs_chain</td>
<td>12</td>
</tr>
<tr>
<td>ggs_compare_partial</td>
<td>13</td>
</tr>
<tr>
<td>ggs_crosscorrelation</td>
<td>14</td>
</tr>
<tr>
<td>ggs_density</td>
<td>15</td>
</tr>
<tr>
<td>ggs_diagnostics</td>
<td>16</td>
</tr>
<tr>
<td>ggs_effective</td>
<td>17</td>
</tr>
<tr>
<td>ggs_geweke</td>
<td>19</td>
</tr>
<tr>
<td>ggs_histogram</td>
<td>20</td>
</tr>
<tr>
<td>ggs_pairs</td>
<td>21</td>
</tr>
<tr>
<td>ggs_pcp</td>
<td>22</td>
</tr>
<tr>
<td>ggs_ppmean</td>
<td>23</td>
</tr>
<tr>
<td>ggs_ppsd</td>
<td>24</td>
</tr>
<tr>
<td>ggs_Rhat</td>
<td>25</td>
</tr>
<tr>
<td>ggs_rocplot</td>
<td>26</td>
</tr>
<tr>
<td>ggs_running</td>
<td>27</td>
</tr>
<tr>
<td>ggs_separation</td>
<td>28</td>
</tr>
<tr>
<td>ggs_traceplot</td>
<td>29</td>
</tr>
<tr>
<td>gl_unq</td>
<td>30</td>
</tr>
<tr>
<td>linear</td>
<td>31</td>
</tr>
<tr>
<td>plab</td>
<td>32</td>
</tr>
<tr>
<td>radon</td>
<td>33</td>
</tr>
<tr>
<td>roc_calc</td>
<td>34</td>
</tr>
<tr>
<td>s</td>
<td>34</td>
</tr>
<tr>
<td>s.binary</td>
<td>35</td>
</tr>
<tr>
<td>s.y.rep</td>
<td>35</td>
</tr>
<tr>
<td>sde0f</td>
<td>36</td>
</tr>
<tr>
<td>y</td>
<td>36</td>
</tr>
<tr>
<td>y.binary</td>
<td>37</td>
</tr>
</tbody>
</table>

**Index**

38
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

\[ \text{ac}(x, \text{nLags}) \]

Arguments

- \( x \) Vector with a chain of simulated values.
- \( \text{nLags} \) Numerical value with the maximum number of lags to take into account.

Value

A matrix with the autocorrelations of every chain.

References

Fernández-i-Marín, Xavier (2016) ggmcmc: Analysis of MCMC Samples and Bayesian Inference. Journal of Statistical Software, 70(9), 1-20. doi:10.18637/jss.v070.i09 Internal function used by \text{ggs\_autocorrelation}.

Examples

\[
\begin{align*}
\text{# Calculate the autocorrelation of a simple vector} \\
\text{ac(cumsum(rnorm(10))/10, nLags=4)}
\end{align*}
\]

Simulated data for a binary logistic regression and its MCMC samples

Description

Simulate a dataset with one explanatory variable and one binary outcome variable using \( y \sim \text{dbern}(\mu); \logit(\mu) = \theta[1] + \theta[2] \times X \). The data loads two objects: the observed \( y \) values and the \text{coda} object containing simulated values from the posterior distribution of the intercept and slope of a logistic regression. The purpose of the dataset is only to show the possibilities of the \text{ggmcmc} package.

Usage

data(binary)
Format

Two objects, namely:

**s.binary** A coda object containing posterior distributions of the intercept (theta[1]) and slope (theta[2]) of a logistic regression with simulated data.

**y.binary** A numeric vector containing the observed values of the outcome in the binary regression with simulated data.

Source

Simulated data for ggmcmc

Examples

data(binary)
str(s.binary)
str(y.binary)
table(y.binary)

---

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

**Arguments**
- **D**  
  Data frame with 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 credible intervals (thin and thick), ready to be used to produce caterpillar plots.

**Examples**
```r
data(linear)
ci(ggs(s))
```

---

**custom.sort**  
*Auxiliary function that sorts Parameter names taking into account numeric values*

**Description**
Auxiliary function that sorts Parameter names taking into account numeric values.

**Usage**
```r
custom.sort(x)
```

**Arguments**
- **x**  
  a character vector to which we want to sort elements
Value

X a character vector sorted with family parameters first and then numeric values

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.

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.
Usage

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

Arguments

- `D` Data frame with 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. When the file has an html file extension the output is an Rmarkdown report with the figures embedded in the html file.
- `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_effective()`, `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 trace-plots. It is not a thin operation, because it is not regular. It must be used with care.
- `dev_type_html` Character. Character vector indicating the type of graphical device for the html output. By default, png. See RMarkdown.
- `...` 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], ...).
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

```r
ggs(
  S,
  family = NA,
  description = NA,
  burnin = TRUE,
  par_labels = NA,
  sort = TRUE,
  splitting = FALSE,
  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 `stanreg` object with samples from rstanarm, a `brmsfit` object with samples from brms, 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 columns. 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.

sort
Logical. When TRUE (the default), parameters are sorted first by family name and then by numerical value.

splitting
Logical. When TRUE, use the approach suggested by Gelman, Carlin, Stern, Dunson, Vehtari and Rubin (2014) Bayesian Data Analysis. 3rd edition. This implies splitting the sequences (original chains) in half, and treat each half as a different Chain, therefore effectively doubling the number of chains. In this case, the first half of Chain 1 is still Chain 1, but the second half is turned into Chain 2, and the first half of Chain 2 into Chain 3, and so on. Defaults to FALSE.

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
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.

References

Examples

# Assign 'S' 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

```r
ggs_autocorrelation(D, family = NA, nLags = 50, greek = 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).
- `nLags`  
  Integer indicating the number of lags of the autocorrelation plot.
- `greek`  
  Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

Value

A `ggplot` object.

Examples

```r
data(linear)
ggs_autocorrelation(ggs(s))
```
**Description**

Caterpillar plots are plotted combining all chains for each parameter.

**Usage**

```r
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,
  label = NULL,
  comparison = NULL,
  comparison_separation = 0.2,
  greek = FALSE,
  sort = TRUE
)
```

**Arguments**

- **D**
  Data frame with 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.

label  Character value with the name of the variable that contains the labels displayed in the plot. Defaults to NULL, which corresponds to using the Parameter name or the Label in case par_labels is used in the ggs() object.

comparison  Character value with the name of the variable that contains the focus of the comparison. Defaults to NULL, which corresponds to no comparison. It is not expected to be used together with X.

comparison_separation  Numerical value with the separation between the dodged parameters. Defaults to 0.2.

greek  Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

sort  Logical value indicating whether, in a horizontal display, y-axis labels must be sorted (the default) or not.

Value

A ggplot object.

References


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
**ggs_compare_partial**

**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**

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

**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).

- **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.

- **greek**  
  Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

**Value**

A ggplot object.

**References**


**Examples**

```r
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, greek = 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).
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.
greek            Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

Value

a ggplot object.

References


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**

```r
ggs_density(D, family = NA, rug = FALSE, hpd = FALSE, greek = FALSE)
```

**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).
- **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.
- **hpd** Logical indicating whether HPD intervals (using the defaults from ci()) must be added to the plot. It is FALSE by default.
- **greek** Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

**Value**

A ggplot object.

**References**


**Examples**

```r
data(linear)
ggs_density(ggs(s))
```
ggs_diagnostics  Formal diagnostics of convergence and sampling quality

Description

Get in a single tidy dataframe the results of the formal (non-visual) convergence analysis. Namely, the Geweke diagnostic (z, from ggs_geweke()), the Potential Scale Reduction Factor Rhat (Rhat, from ggs_Rhat()) and the number of effective independent draws (Effective, from ggs_effective()).

Usage

```r
ggs_diagnostics(
  D,
  family = NA,
  version_rhat = "BDA2",
  version_effective = "spectral",
  proportion = TRUE
)
```

Arguments

- **D**: Data frame with the simulations
- **family**: Name of the family of parameters to return, 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).
- **version_rhat**: Character variable with the name of the version of the potential scale reduction factor to use. Defaults to "BDA2", which refers to the second version of _Bayesian Data Analysis_ (Gelman, Carlin, Stern and Rubin). The other available version is "BG98", which refers to Brooks & Gelman (1998) and is the one used in the "coda" package.
- **version_effective**: Character variable with the name of the version of the calculation to use. Defaults to "spectral", which refers to the simple version estimating the spectral density at frequency zero used in the "coda" package. An alternative version "BDA3" is provided, which refers to the third edition of Bayesian Data Analysis (Gelman, Carlin, Stern, Dunson, Vehtari and Rubin).
- **proportion**: Logical value whether to return the proportion of effective independent draws over the total (the default) or the number.

Details

Notice that at least two chains are required. Otherwise, only the Geweke diagnostic makes sense, and can be returned with its own function.
**Value**

A tidy dataframe.

**References**


**See Also**

`ggs_geweke`, `ggs_Rhat` and `ggs_effective` for their respective options.

**Examples**

```r
data(linear)
ggs_diagnostics(ggs(s))
```

---

**ggs_effective**

**Dotplot of the effective number of independent draws**

**Description**

Dotplot of the effective number of independent draws. The default version is the sample size adjusted for autocorrelation. An alternative from the third edition of Bayesian Data Analysis (Gelman, Carlin, Stern, Dunson, Vehtari and Rubin) is provided.

**Usage**

```r
ggs_effective(
  D,
  family = NA,
  greek = FALSE,
  version_effective = "spectral",
  proportion = TRUE,
  plot = TRUE
)
```
**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).

- **greek**  
Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

- **version_effective**  
Character variable with the name of the version of the calculation to use. Defaults to "spectral", which refers to the simple version estimating the spectral density at frequency zero used in the "coda" package. An alternative version "BDA3" is provided, which refers to the third edition of Bayesian Data Analysis (Gelman, Carlin, Stern, Dunson, Vehtari and Rubin).

- **proportion**  
Logical value whether to return the proportion of effective independent draws over the total (the default) or the number.

- **plot**  
Logical value indicating whether the plot must be returned (the default) or a tidy dataframe with the effective number of samples per Parameter.

**Details**

Notice that at least two chains are required.

**Value**

A ggplot object, or a tidy data frame.

**References**


**Examples**

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

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

**Description**

Dotplot of Geweke diagnostic.

**Usage**

```r
ggs_geweke(
  D,
  family = NA,
  frac1 = 0.1,
  frac2 = 0.5,
  shadow_limit = TRUE,
  greek = FALSE,
  plot = TRUE
)
```

**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).

- **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.

- **greek**,  
  Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

- **plot**,  
  Logical value indicating whether the plot must be returned (the default) or a tidy dataframe with the results of the Geweke diagnostics per Parameter and Chain.

**Value**

A ggplot object, or a tidy data frame.

**References**


**ggs_histogram**

**Examples**

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

---

**ggs_histogram**  
Histograms of the parameters.

**Description**

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

**Usage**

```r
ggs_histogram(D, family = NA, bins = 30, greek = FALSE)
```

**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).
- **bins**: Integer indicating the total number of bins in which to divide the histogram. Defaults to 30, which is the same as geom_histogram().
- **greek**: Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

**Value**

A ggplot object.

**References**


**Examples**

```r
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, greek = FALSE, ...)

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).

greek
Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

...
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.

References

Examples
## Not run:
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)))

## End(Not run)

---

**ggs_pcp**

*Plot for model fit of binary response variables: percent correctly predicted*

**Description**

Plot a histogram with the distribution of correctly predicted cases in a model against a binary response variable.

**Usage**

```r
Ggs_pcp(D, outcome, threshold = "observed", bins = 30)
```

**Arguments**

- **D**: Data frame with 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.
- **threshold**: numerical bounded between 0 and 1 or "observed", the default. If "observed", the threshold of expected values to be considered a realization of the event (1, success) is computed using the observed value in the data. Otherwise, a numerical value showing which threshold to use (typically, 0.5) can be given.
- **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**

```r
data(binary)
ggs_pcp(ggs(s.binary, family="mu"), outcome=y.binary)
```
**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**

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

**Arguments**

- `D`: Data frame with 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.

**References**


**Examples**

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

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

**Usage**

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

**Arguments**

- `D`: Data frame with 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.

**References**


**Examples**

```r
data(linear)
ggs_ppsd(ggs(s.y.rep), outcome=y)
```
### 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, Stern and Rubin) is used, but the version used in the package “coda” can also be used (Brooks & Gelman 1998).

### Usage

```r
ggplot_Rhat(
  D,
  family = NA,
  scaling = 1.5,
  greek = FALSE,
  version_rhat = "BDA2",
  plot = TRUE
)
```

### 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).
- **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.
- **greek**: Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.
- **version_rhat**: Character variable with the name of the version of the potential scale reduction factor to use. Defaults to "BDA2", which refers to the second version of _Bayesian Data Analysis_ (Gelman, Carlin, Stern and Rubin). The other available version is "BG98", which refers to Brooks & Gelman (1998) and is the one used in the “coda” package.
- **plot**: Logical value indicating whether the plot must be returned (the default) or a tidy dataframe with the results of the Rhat diagnostics per Parameter.

### Details

Notice that at least two chains are required.

### Value

A ggplot object, or a tidy data frame.
References


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 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

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

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.
- **greek**: Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.

Value

A ggplot object.

References


Examples

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

Plot a separation plot with the results of the model against a binary response variable.

Usage

```r
ggs_separation(
  D,
  outcome,
  minimalist = FALSE,
  show_labels = FALSE,
  uncertainty_band = TRUE
)
```

Arguments

- **D**: Data frame with 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.
- **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.
- **show_labels**: logical, FALSE by default. If TRUE it adds the Parameter as the label of the case in the x-axis.
- **uncertainty_band**: logical, TRUE by default. If FALSE it removes the uncertainty band on the predicted values.

Value

A ggplot object

References


**Examples**

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

---

**Description**

Traceplot with the time series of the chains.

**Usage**

```r
ggs_traceplot(
  D,
  family = NA,
  original_burnin = TRUE,
  original_thin = TRUE,
  simplify = NULL,
  hpd = FALSE,
  greek = FALSE
)
```

**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 traceplots. It is not a thin operation, because it is not regular. It must be used with care.

- `hpd`  
  Logical indicating whether HPD intervals (using the defaults from ci()) must be added to the plot. It is FALSE by default.

- `greek`  
  Logical value indicating whether parameter labels have to be parsed to get Greek letters. Defaults to false.
Value

A ggplot object.

References


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

```r
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
Simulated data for a continuous linear regression and its MCMC samples

Description

Simulate a dataset with one explanatory variable and one continuous outcome variable using \((y \sim \text{dnorm}(\mu, \sigma); \mu = \beta[1] + \beta[2] \times X)\). The data loads three objects: the observed \(y\) values, a coda object containing simulated values from the posterior distribution of the intercept and slope of a linear regression, and a coda object containing simulated values from the posterior predictive distribution. The purpose of the dataset is only to show the possibilities of the ggmcmc package.

Usage

data(linear)

Format

Three objects, namely:

- **s**: A coda object containing posterior distributions of the intercept (\(\beta[1]\)) and slope (\(\beta[2]\)) of a linear regression with simulated data.
- **s.y.rep**: A coda object containing simulated values from the posterior predictive distribution of the outcome of a linear regression with simulated data \((y \sim \text{N}(\mu, \sigma); \mu = \beta[1] + \beta[2] \times X; \text{y.rep} \sim \text{N}(\mu, \sigma)); \text{where y.rep is a replicated outcome, originally missing data}.)
- **y**: A numeric vector containing the observed values of the outcome in the linear regression with simulated data.

Source

Simulated data for ggmcmc

Examples

data(linear)
str(s)
str(s.y.rep)
str(y)
plab

Generate a data frame suitable for matching parameter names with their labels

Description

Generate a data frame with at least columns for Parameter and Labels. This function is intended to work as a shortcut for the matching data frame necessary to pass the argument "par_labels" to ggs() calls for transforming the parameter names.

Usage

plab(parameter.name, match, subscripts = NULL)

Arguments

parameter.name A character vector of length one with the name of the variable (family) without subscripts. Usually, it refers to a Greek letter.

match A named list with the variable labels and the values of the factor corresponding to the dimension they map to. The order of the list matters, as ggmcmc assumes that the first dimension corresponds to the first element in the list, and so on.

subscripts An optional character with the letters that correspond to each of the dimensions of the family of parameters. By default it uses not very informative names "dim.1", "dim.2", etc... It usually corresponds to the "i", "j", ... subscripts in classical textbooks, but is recommended to be closer to the subscripts given in the sampling software.

Value

A data frame tbl with the Parameter names and its match with meaningful variable Labels. Also the intermediate variables are passed to make it easier to work with the samples using meaningful variable names.

Examples

data(radon)
L.radon <- plab("alpha", match = list(County = radon$counties$County))
# Generates a data frame suitable for matching with the generated samples
# through the "par_labels" function:
ggs_caterpillar(ggs(radon$s.radon, par_labels = L.radon, family = "alpha"))
Simulations of the parameters of a hierarchical model

Description

Using the radon example in Gelman & Hill (2007), the list contains several elements to show the possibilities of ggmcmc for applied Bayesian Hierarchical/multilevel analysis.

Usage

data(radon)

Format

A list containing several elements (data and outputs of the analysis):

- **counties** A data frame with the country 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.
- **s.radon.short** A coda object with simulated values from the posterior distribution of few parameters, with reasonable chain length.

Source

http://www.stat.columbia.edu/~gelman/arm/examples/radon/

Examples

data(radon)
names(radon)

# Generate a data frame suitable for matching with the generated samples
# through the "par_labels" function:
L.radon <- plab("alpha", match = list(County = radon$counties$County))
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.

**References**

---

*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**
data(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 \((y \sim \text{dbern}(mu); \text{logit}(mu) = \theta[1] + \theta[2] \times X)\), and the fitted / expected values \((mu)\). The purpose of the dataset is only to show the possibilities of the ggmcmc package.

**Usage**

```r
data(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 \((y \sim \text{N}(mu, sigma); \ mu = \beta[1] + \beta[2] \times X; y\text{.rep} \sim \text{N}(mu, sigma));\) where \(y\text{.rep}\) is a replicated outcome, originally missing data). The purpose of the dataset is only to show the possibilities of the ggmcmc package.

**Usage**

```r
data(s.y.rep)
```

**Format**

A coda object containing posterior distributions of the posterior predictive distribution of a linear regression with fake data.
\textbf{Description}

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

\textbf{Usage}

\texttt{sde\_0f(x)}

\textbf{Arguments}

\texttt{x} \hspace{1cm} A time series

\textbf{Details}

Internal function to compute the Spectral Density Estimate at Zero Frequency for a given chain used by \texttt{ggs\_geweke}.

\textbf{Value}

A vector with the spectral density estimate at zero frequency

\textbf{Description}

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

\textbf{Usage}

\texttt{data(y)}

\textbf{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
data(y.binary)

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

*Topic datasets
binary, 3
linear, 31
radon, 33
s, 34
s.binary, 35
s.y.rep, 35
y, 36
y.binary, 37
ggs_traceplot, 29
g1_unq, 30
linear, 31
plab, 32
radon, 33
roc_calc, 34
s, 34
s.binary, 35
s.y.rep, 35
sde0f, 36
y, 36
y.binary, 37
ac, 3
binary, 3
calc_bin, 4
ci, 5
custom.sort, 5
get_family, 6
ggmcmc, 6
ggmcmc-package (ggmcmc), 6
...