Package: bayesGAM (via r-universe)

August 27, 2024

Title Fit Multivariate Response Generalized Additive Models using Hamiltonian Monte Carlo

Version 0.0.2

Description The 'bayesGAM' package is designed to provide a user friendly option to fit univariate and multivariate response Generalized Additive Models (GAM) using Hamiltonian Monte Carlo (HMC) with few technical burdens. The functions in this package use 'rstan' (Stan Development Team 2020) to call 'Stan' routines that run the HMC simulations. The 'Stan' code for these models is already pre-compiled for the user. The programming formulation for models in 'bayesGAM' is designed to be familiar to analysts who fit statistical models in 'R'. Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., ... & Riddell, A. (2017). Stan: A probabilistic programming language. Journal of statistical software, 76(1). Stan Development Team. 2018. RStan: the R interface to Stan. R package version 2.17.3. <https://mc-stan.org/> Neal, Radford (2011) ``Handbook of Markov Chain Monte Carlo" ISBN: 978-1420079418. Betancourt, Michael, and Mark Girolami. ``Hamiltonian Monte Carlo for hierarchical models." Current trends in Bayesian methodology with applications 79.30 (2015): 2-4. Thomas, S., Tu, W. (2020) ``Learning Hamiltonian Monte Carlo in R" <arXiv:2006.16194>, Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013) ``Bayesian Data Analysis" ISBN: 978-1439840955, Agresti, Alan (2015) ``Foundations of Linear and Generalized Linear Models ISBN: 978-1118730034, Pinheiro, J., Bates, D. (2006) "Mixed-effects Models in S and S-Plus" ISBN: 978-1441903174. Ruppert, D., Wand, M. P., & Carroll, R. J. (2003). Semiparametric regression (No. 12). Cambridge university press. ISBN: 978-0521785167.

Maintainer Samuel Thomas <samthoma@alumni.iu.edu>

Depends R (>= 3.6)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.2

Biarch true

Imports bayesplot, boot, cluster, corrplot, ggplot2, graphics, gridExtra, loo, methods, mlbench, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), rstantools (>= 2.1.0.9000), SemiPar, stats, geometry, MASS

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

Suggests testthat

SystemRequirements GNU make

NeedsCompilation yes

Author Samuel Thomas [cre, aut], Wanzhu Tu [ctb], Trustees of Columbia University (R/rstanMethods.R) [cph]

Repository CRAN

Date/Publication 2022-03-17 08:30:06 UTC

Contents

---

bayesGAM-package *The 'bayesGAM' package.*

Description

Fit Bayesian multivariate generalized additive models using Stan

References

Stan Development Team (2020). RStan: the R interface to Stan. R package version 2.21.1. https://mc-stan.org

---

bayesGAM bayesGAM *fits a variety of regression models using Hamiltonian Monte Carlo*

Description

Based on glm. bayesGAM is used to fit a variety of statistical models, including linear models, generalized lienar models, mixed effect models with random intercept, and semiparametric regression models.

Usage

```
bayesGAM(
  formula,
  random = NULL,
  family = gaussian,
  data,
  offset,
  beta = list(),
  eps = list(),
  lambda = list(),
  a = list(),
  spcontrol = list(qr = TRUE, mvindep = FALSE, ...),
  store_plot_data = FALSE,
  method = "bayesGAMfit",
  ...
)
```

Arguments

formula	a formula object describing the model to be fitted.
random	(optional) specify a random intercept in the form '~var'
family	distribution and link function for the model
data	(optional) data frame containing the variables in the model.
offset	Same as glm
beta	(optional) list of priors for the fixed effects parameters. Sensible priors are selected as a default.
eps	(optional) list of priors for the error term in linear regression. Sensible priors are selecteda as a default.
lambda	(optional) list of priors for random effects variance parameters. Sensible priors are selected as a default.
a	(optional) list of priors for the off diagonal of the LDLT decomposed covariance matrix for multivariate response models. Vague normal priors are used as a default.
spcontrol	a list of control parameters for fitting the model in STAN. See 'details'
store_plot_data	a logical indicator for storing the plot data frame after simulation. Defaults to FALSE
method	default currently set to 'bayesGAMfit'.
...	Arguments passed to rstan::sampling (e.g. iter, chains).

Details

Similar to glm, models are typically specified by formula. The formula typically takes the form response ~ terms, where the response is numeric and terms specify the linear predictor for the response. The terms may be numeric variables or factors.

The link function for the Generalized Linear Model is specified with a family object. Currently, this package supports gaussian, binomial, and poisson families with all available link functions.

The list spcontrol currently supports additional parameters to facilitate fitting models. qr is a logical indicator specifying whether the design matrix should be transformed via QR decomposition prior to HMC sampling. QR decomposition often improves the efficiency with which HMC samples, as the MCMC chain navigates an orthogonal space more easily than highly correlated parameters. mvindep is a logical indicator for multivariate response models with random intercepts. This indicates whether the multivariate responses should be considered independent. Defaults to FALSE

Value

An object of class bayesGAMfit. Includes slots:

results: stanfit object returned by rstan::sampling

model: glmModel object

offset: offset vector from the input parameter

spcontrol: list of control parameters from input

References

Hastie, T. J. (1992) Generalized additive models. Chapter 7 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

Dobson, A. J. (1990) *An Introduction to Generalized Linear Models*. London: Chapman and Hall.

Examples

```
## Dobson (1990) Page 93: Randomized Controlled Trial :
counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)
fpois<- bayesGAM(counts ~ outcome + treatment, family = poisson(),
                 spcontrol = list(qr = TRUE))
summary(fpois)
```

---

bayesGAMfit-class *Contains results from* rstan *as well as the design matrices and other data for the model.*

Description

Returns object generated from model fit by bayesGAM

Usage

```
## S4 method for signature 'bayesGAMfit'
show(object)
```

Arguments

object	Object of type bayesGAMfit which stores the results from rstan, design matrices, and other data for the model.

Slots

results Object of type stanfit returned from calling rstan::sampling

model Object of custom type glmModel with the data and input parameters passed to rstan

offset Optionally numeric offset for the generalized additive model

spcontrol List of control parameters for bayesGAMfit

mcmcres Matrix of MCMC results for all chains, if plot data is stored

pdata Dataframe for default plot method, if plot data is stored.

`bloodpressure` *Blood pressure data from a clinical study*

Description

Data from 200 subjects

Usage

```
bloodpressure
```

Format

A data frame with 2438 rows and 13 variables:

ID Subject identification number

BIRTH_WT birth weight (lbs)

WEIGHT current weight (lbs)

HEIGHT current height (cm)

BMI current body mass index

age current age (yrs)

dias diastolic blood pressure

sys systolic blood pressure

SexM indicator of sex male

RaceB indicator of race black

RaceW indicator of race white

PHIGHBP indicator that either parent had high blood pressure

PDIABET indicator that either parent had diabetes

Source

Data provided by Wanzhu Tu, Indiana University School of Medicine

References

Tu, W., Eckert, G. J., DiMeglio, L. A., Yu, Z., Jung, J., and Pratt, J. H. (2011). *Intensified effect of adiposity on blood pressure in overweight and obese children*. Hypertension, 58(5), 818-824.

coefficients *Extract Model Coefficients*

Description

Method for bayesGAMfit objects. Extracts the specified quantile of the posterior. The user may specify all or some of the parameters β, ϵ, λ, u, $sigma$, a.

Usage

```
## S4 method for signature 'bayesGAMfit'
coefficients(
  object,
  params = c("beta", "eps", "lambda", "u", "sigma", "a"),
  FUN = median
)

## S4 method for signature 'bayesGAMfit'
coef(
  object,
  params = c("beta", "eps", "lambda", "u", "sigma", "a"),
  FUN = median
)
```

Arguments

object	an object of class bayesGAMfit, usually a result of a call to bayesGAM.
params	character vector of the names of parameters to return • β beta • ϵ eps • λ lambda • $a]a$
FUN	function from which to estimate coefficients. Default is median

Value

Numeric vector of parameter point estimates based on the given prob, with a default of the median estimate.

Examples

```
require(stats); require(graphics)
f <- bayesGAM(weight ~ np(height), data = women, family = gaussian,
              iter = 500, chains = 1)
coef(f, params=c("beta", "eps"))
```

create_bivariate_design *Creates a design matrix from a bivariate smoothing algorithm*

Description

create_bivariate_design accepts two numeric vectors of equal length as inputs. From these inputs, a bivariate smoothing design matrix is produced using thin plate splines.

Usage

```
create_bivariate_design(X1, X2, num_knots = NULL, knots = NULL)
```

Arguments

X1	numeric vector for first variable
X2	numeric vector for second variable
num_knots	optional: number of knots
knots	optional: matrix of knot locations for bivariate smoothing

Value

list containing the design matrix Z and matrix of knots

References

Ruppert, David, Matt P. Wand, and Raymond J. Carroll. *Semiparametric Regression*. No. 12. Cambridge university press, 2003. Section 13.5

Matt Wand (2018). SemiPar: Semiparametric Regression. R package version 1.0-4.2.

Examples

```
x1 <- rnorm(100)
x2 <- rnorm(100)
res <- create_bivariate_design(x1, x2)
res$knots
dim(res$Z)
```

`extract_log_lik_bgam` *Extract the log likelihood from models fit by* `bayesGAM`

Description

Convenience function for extracting the pointwise log-likelihood matrix or array from a model fit by `bayesGAM`. Calls the `extract_log_lik` method from the `loo` package

Usage

```
extract_log_lik_bgam(object, ...)

## S4 method for signature 'bayesGAMfit'
extract_log_lik_bgam(object, ...)
```

Arguments

`object`	Object of type `bayesGAMfit` generated from `bayesGAM`.
`...`	Additional parameters to pass to `loo::extract_log_lik`

Value

A matrix with the extracted log likelihood values post-warmup

References

Stan Development Team (2017). The Stan C++ Library, Version 2.16.0. https://mc-stan.org/

Stan Development Team (2017). RStan: the R interface to Stan, Version 2.16.1. https://mc-stan.org/

Vehtari A, Gabry J, Magnusson M, Yao Y, Gelman A (2019). "loo: Efficient leave-one-out cross-validation and WAIC for Bayesian models." R package version 2.2.0, <URL: https://mc-stan.org/loo>.

Vehtari A, Gelman A, Gabry J (2017). "Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC." *Statistics and Computing*, *27*, 1413-1432. doi:10.1007/s11222-016-9696-4 (URL: https://doi.org/10.1007/s11222-016-9696-4).

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=500, chains = 1)
ll <- extract_log_lik_bgam(f)
```

fitted *Extract fitted values from a model fit by* bayesGAM

Description

Method for bayesGAMfit objects. Extracts the fitted values based on a specified quantile for the posterior distribution. The median is the default.

Usage

```
## S4 method for signature 'bayesGAMfit'
fitted(object, ...)
```

Arguments

object	an object of class bayesGAMfit, usually a result of a call to bayesGAM.
...	additional arguments to pass to coefficients

Value

Numeric vector of fitted values

Examples

```
require(stats); require(graphics)
f <- bayesGAM(weight ~ np(height), data = women, family = gaussian,
              iter = 500, chains = 1)
plot(fitted(f), women$weight, type='o', xlab="fitted", ylab="actual")
```

getDesign *Design matrices from a* bayesGAMfit *object*

Description

Contains the design matrices produced for model fitting. The fixed effects design matrix X or random effects design matrix Z can be specified.

Usage

```
getDesign(object, ...)

## S4 method for signature 'bayesGAMfit'
getDesign(object, type = "X")

## S4 method for signature 'glmModel'
getDesign(object, type = "X")
```

Arguments

object	Object of type bayesGAMfit generated from bayesGAM
...	Additional arguments for getDesign
type	Character for fixed effect design matrix X or random effects design matrix Z

Value

Contents of stanfit results

Examples

```
require(stats); require(graphics)
f <- bayesGAM(weight ~ np(height), data = women, family = gaussian,
              iter = 500, chains = 1)
getDesign(f, "Z")
```

---

getModelSlots *Return one or slots from the* Stan *model in* bayesGAM

Description

Contains the objects and parameters passed to Stan in object of type glmModel, contained in object type bayesGAMfit

Usage

```
getModelSlots(object, ...)

## S4 method for signature 'bayesGAMfit'
getModelSlots(object, name = "X")
```

Arguments

object	Object of type bayesGAMfit
...	Additional arguments for getModelSlots
name	Character name of slot in glmModel

- X Fixed effects design matrix
- Z Random effects design matrix
- Zlst list of individual random effects design matrices that, combined, form Z
- Zarray array of individual random effects design matrices. Used for multiple response models
- max_col maximum number of columns of an individual Z matrix. Padding for STAN
- y numeric response matrix

- p number of beta parameters
- r number of eps parameters
- q number of lambda parameters
- n number of records in the dataset
- has_intercept logical of whether the model includes an intercept term
- zvars number of random effects variables
- names_beta parameter names for beta
- names_u parameter names for the random effects
- names_y response names
- prior `prior` object with priors used in the model
- knots list of knots used in non-parametric functions
- basis character indicating basis function. `tps` for thin-plate splines and `trunc.poly` for truncated polynomial
- npargs arguments passed to non-parametric functions in the model
- npterms variables used in non-parametric functions
- sub_form formula with the np terms removed
- random_intercept logical indicator of whether a random effects intercept is used
- multresponse logical indicator of whether the model is multiple response

Value

Contents of slot in `glmModel`

Examples

```
require(stats); require(graphics)
f <- bayesGAM(weight ~ np(height), data = women, family = gaussian,
              iter = 500, chains = 1)
getModelSlots(f, "X")
```

`getSamples` *Extract the MCMC samples from an object of type* `bayesGAMfit`

Description

Returns an array of the posterior simulation from `Stan`. Optionally, may return a subsample from the full MCMC simulation.

Usage

```
getSamples(object, ...)

## S4 method for signature 'bayesGAMfit'
getSamples(object, nsamp = NULL, seednum = NULL, ...)

## S4 method for signature 'stanfit'
getSamples(object, nsamp = 1000, seednum = NULL, results = NULL, ...)

## S4 method for signature 'glmModel'
getSamples(object, nsamp = NULL, seednum = NULL, results = NULL, ...)
```

Arguments

object	model object of class bayesGAMfit
...	Additional parameters passed to corrplot.mixed
nsamp	Optional number of samples to return
seednum	Optional integer for seed number when selecting a random sample
results	Matrix of HMC posterior samples

Value

array of the posterior simulation, or subsample of the array

NA

Examples

```
require(stats); require(graphics)
f <- bayesGAM(weight ~ np(height), data = women, family = gaussian,
              iter = 500, chains = 1)
allres <- getSamples(f)
```

---

getStanResults *Returns the* stanfit *object generated by* **rstan**

Description

Contains the full content of the stanfit object

Usage

```
getStanResults(object)

## S4 method for signature 'bayesGAMfit'
getStanResults(object)
```

Arguments

object	Object of type bayesGAMfit returned from bayesGAM

Value

Contents of stanfit results

Examples

```
require(stats); require(graphics)
f <- bayesGAM(weight ~ np(height), data = women, family = gaussian,
              iter = 500, chains = 1)
sres <- getStanResults(f)
plot(sres) # rstan method
```

L *Lag function for autoregressive models*

Description

Creates lagged variables for use with bayesGAM, including the functionality to create lags for each specified subject if desired. The input data must be pre- sorted according by time, and within each subject id if specified.

Usage

```
L(x, k = 1, id = NULL)
```

Arguments

x	numeric vector
k	integer vector of lagged variables to create
id	optional identification number for each subject

Value

numeric vector or matrix of the lagged variable(s)

References

Zeileis A (2019). dynlm: Dynamic Linear Regression. R package version 0.3-6

Examples

```
x <- rnorm(20)
id <- rep(1:4, each=5)
L(x, 1:2, id)

# autoregressive
ar.ols(lh, demean = FALSE, intercept=TRUE, order=1)
f <- bayesGAM(lh ~ L(lh), family=gaussian)
coef(f)
```

`loo_bgam` *Calls the* `loo` *package to perform efficient approximate leave-one-out cross-validation on models fit with* `bayesGAM`

Description

Computes PSIS-LOO CV, efficient approximate leave-one-out (LOO) cross-validation for Bayesian models using Pareto smoothed importance sampling (PSIS). This calls the implementation from the `loo` package of the methods described in Vehtari, Gelman, and Gabry (2017a, 2017b).

Usage

```
loo_bgam(object, ...)

## S4 method for signature 'bayesGAMfit'
loo_bgam(object, ...)

## S4 method for signature 'array'
loo_bgam(object, ...)
```

Arguments

`object`	Object of type `bayesGAMfit` generated from `bayesGAM`.
`...`	Additional parameters to pass to pass to `loo::loo`

Value

a named list of class `c("psis_loo", "loo")`

`estimates` A matrix with two columns (`Estimate`, `SE`) and three rows (`elpd_loo`, `p_loo`, `looic`). This contains point estimates and standard errors of the expected log pointwise predictive density (`elpd_loo`), the effective number of parameters (`p_loo`) and the LOO information criterion `looic` (which is just `-2 * elpd_loo`, i.e., converted to deviance scale).

`pointwise` A matrix with five columns (and number of rows equal to the number of observations) containing the pointwise contributions of the measures (`elpd_loo`, `mcse_elpd_loo`, `p_loo`, `looic`, `influence_pareto_k`). in addition to the three measures in `estimates`, we also report

pointwise values of the Monte Carlo standard error of `elpd_loo` (`mcse_elpd_loo`), and statistics describing the influence of each observation on the posterior distribution (`influence_pareto_k`). These are the estimates of the shape parameter k of the generalized Pareto fit to the importance ratios for each leave-one-out distribution. See the pareto-k-diagnostic page for details.

`diagnostics` A named list containing two vectors:

- `pareto_k`: Importance sampling reliability diagnostics. By default, these are equal to the `influence_pareto_k` in `pointwise`. Some algorithms can improve importance sampling reliability and modify these diagnostics. See the pareto-k-diagnostic page for details.
- `n_eff`: PSIS effective sample size estimates.

`psis_object` This component will be `NULL` unless the `save_psis` argument is set to `TRUE` when calling `loo()`. In that case `psis_object` will be the object of class `"psis"` that is created when the `loo()` function calls `psis()` internally to do the PSIS procedure.

References

Vehtari, A., Gelman, A., and Gabry, J. (2017a). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. Statistics and Computing. 27(5), 1413–1432. doi:10.1007/s11222-016-9696-4 (journal version, preprint arXiv:1507.04544).

Vehtari, A., Gelman, A., and Gabry, J. (2017b). Pareto smoothed importance sampling. preprint arXiv:1507.02646

Vehtari A, Gabry J, Magnusson M, Yao Y, Gelman A (2019). "loo: Efficient leave-one-out cross-validation and WAIC for Bayesian models." R package version 2.2.0, <URL: https://mc-stan.org/loo>.

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=500, chains = 1)
loo_bgam(f)
```

`loo_compare_bgam` *Calls the* `loo` *package to compare models fit by* `bayesGAMfit`

Description

Compares fitted models based on ELPD, the expected log pointwise predictive density for a new dataset.

Usage

```
loo_compare_bgam(object, ...)

## S4 method for signature 'bayesGAMfit'
loo_compare_bgam(object, ...)
```

Arguments

object	Object of type bayesGAMfit generated from bayesGAM.
...	Additional objects of type bayesGAMfit

Value

a matrix with class compare.loo that has its own print method from the loo package

References

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely application information criterion in singular learning theory. Journal of Machine Learning Research 11, 3571-3594.

Vehtari, A., Gelman, A., and Gabry, J. (2017a). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. Statistics and Computing. 27(5), 1413–1432. doi:10.1007/s11222-016-9696-4 (journal version, preprint arXiv:1507.04544).

Vehtari, A., Gelman, A., and Gabry, J. (2017b). Pareto smoothed importance sampling. preprint arXiv:1507.02646

Vehtari A, Gabry J, Magnusson M, Yao Y, Gelman A (2019). "loo: Efficient leave-one-out cross-validation and WAIC for Bayesian models." R package version 2.2.0, <URL: https://mc-stan.org/loo>.

Gabry, J. , Simpson, D. , Vehtari, A. , Betancourt, M. and Gelman, A. (2019), Visualization in Bayesian workflow. J. R. Stat. Soc. A, 182: 389-402. doi:10.1111/rssa.12378

Examples

```
f1 <- bayesGAM(weight ~ height, data = women,
              family = gaussian, iter=500, chains = 1)
f2 <- bayesGAM(weight ~ np(height), data=women,
              family = gaussian, iter=500, chains = 1)
loo_compare_bgam(f1, f2)
```

---

mcmc_plots *Plotting for MCMC visualization and diagnostics provided by* bayesplot *package*

Description

Plots of Rhat statistics, ratios of effective sample size to total sample size, and autocorrelation of MCMC draws.

Usage

```
mcmc_intervals(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_intervals(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_areas(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_areas(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_hist(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_hist(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_hist_by_chain(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_hist_by_chain(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_dens(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_dens(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_scatter(object, ...)
```

```
## S4 method for signature 'bayesGAMfit'
mcmc_scatter(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_hex(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_hex(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_pairs(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_pairs(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_acf(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_acf(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_acf_bar(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_acf_bar(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_trace(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_trace(
  object,
```

```
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_rhat(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_rhat(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_rhat_hist(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_rhat_hist(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_rhat_data(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_rhat_data(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_neff(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_neff(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_neff_hist(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_neff_hist(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)
```

```
mcmc_neff_data(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_neff_data(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)

mcmc_violin(object, ...)

## S4 method for signature 'bayesGAMfit'
mcmc_violin(
  object,
  regex_pars = c("^beta", "^lambda", "^eps", "^a", "^sigma_u_correlation"),
  ...
)
```

Arguments

`object`	an object of class `bayesGAMfit`
`...`	optional additional arguments to pass to the `bayesplot` functions
`regex_pars`	character vector of regular expressions of variable names to plot

Value

These functions call various plotting functions from the `bayesplot` package, which returns a list including `ggplot2` objects.

Plot Descriptions from the `bayesplot` package documentation

- `mcmc_hist(object, ...)` Default plot called by `plot` function. Histograms of posterior draws with all chains merged.
- `mcmc_dens(object, ...)` Kernel density plots of posterior draws with all chains merged.
- `mcmc_hist_by_chain(object, ...)` Histograms of posterior draws with chains separated via faceting.
- `mcmc_dens_overlay(object, ...)` Kernel density plots of posterior draws with chains separated but overlaid on a single plot.
- `mcmc_violin(object, ...)` The density estimate of each chain is plotted as a violin with horizontal lines at notable quantiles.
- `mcmc_dens_chains(object, ...)` Ridgeline kernel density plots of posterior draws with chains separated but overlaid on a single plot. In `mcmc_dens_overlay()` parameters appear in separate facets; in `mcmc_dens_chains()` they appear in the same panel and can overlap vertically.
- `mcmc_intervals(object, ...)` Plots of uncertainty intervals computed from posterior draws with all chains merged.

- `mcmc_areas(object, ...)` Density plots computed from posterior draws with all chains merged, with uncertainty intervals shown as shaded areas under the curves.
- `mcmc_scatter(object, ...)` Bivariate scatterplot of posterior draws. If using a very large number of posterior draws then `mcmc_hex()` may be preferable to avoid overplotting.
- `mcmc_hex(object, ...)` Hexagonal heatmap of 2-D bin counts. This plot is useful in cases where the posterior sample size is large enough that `mcmc_scatter()` suffers from overplotting.
- `mcmc_pairs(object, ...)` A square plot matrix with univariate marginal distributions along the diagonal (as histograms or kernel density plots) and bivariate distributions off the diagonal (as scatterplots or hex heatmaps).

  For the off-diagonal plots, the default is to split the chains so that (roughly) half are displayed above the diagonal and half are below (all chains are always merged together for the plots along the diagonal). Other possibilities are available by setting the `condition` argument.
- `mcmc_rhat(object, ...)`, `mcmc_rhat_hist(object, ...)` Rhat values as either points or a histogram. Values are colored using different shades (lighter is better). The chosen thresholds are somewhat arbitrary, but can be useful guidelines in practice.
  - *light*: below 1.05 (good)
  - *mid*: between 1.05 and 1.1 (ok)
  - *dark*: above 1.1 (too high)

  `mcmc_neff(object, ...)`, `mcmc_neff_hist(object, ...)` Ratios of effective sample size to total sample size as either points or a histogram. Values are colored using different shades (lighter is better). The chosen thresholds are somewhat arbitrary, but can be useful guidelines in practice. *light*: between 0.5 and 1 (high) *mid*: between 0.1 and 0.5 (good) *dark*: below 0.1 (low)

  `mcmc_acf(object, ...)`, `mcmc_acf_bar(object, ...)` Grid of autocorrelation plots by chain and parameter. The `lags` argument gives the maximum number of lags at which to calculate the autocorrelation function. `mcmc_acf()` is a line plot whereas `mcmc_acf_bar()` is a barplot.

References

Gabry, Jonah and Mahr, Tristan (2019). *bayesplot: Plotting for Bayesian Models*. https://mc-stan.org/bayesplot/

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., and Gelman, A (2019). *Visualization in Bayesian Workflow*. Journal of the Royal Statistical Society: Series A. Vol 182. Issue 2. p.389-402.

Gelman, A. and Rubin, D. (1992) *Inference from Iterative Simulation Using Multiple Sequences*. Statistical Science 7(4) 457-472.

Gelman, A., et. al. (2013) *Bayesian Data Analysis*. Chapman and Hall/CRC.

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=1000, chains = 1)
mcmc_trace(f)
```

mvcorrplot *Multivariate response correlation plot for* bayesGAMfit *objects*

Description

Creates a correlation plot of the multivariate responses based on corrplot

Usage

```
## S4 method for signature 'bayesGAMfit'
mvcorrplot(object, ...)
```

Arguments

object	model object of class bayesGAMfit
...	Additional parameters passed to corrplot.mixed

Value

corrplot object

References

Taiyun Wei and Viliam Simko (2017). R package *corrplot*: Visualization of a Correlation Matrix (Version 0.84).

Examples

```
require(MASS)
sig <- matrix(c(1, 0.5, 0.5, 1), ncol=2)
set.seed(123)
Y <- mvrnorm(50, mu=c(-2, 2), Sigma=sig)
dat <- data.frame(id = rep(1:5, each=10),
                  y1 = Y[, 1],
                  y2 = Y[, 2])

f <- bayesGAM(cbind(y1, y2) ~ 1, random = ~factor(id),
              data=dat,
              a = normal(c(0, 5)),
              chains = 1, iter = 500)
mvcorrplot(f)
```

normal *Constructor function for Normal priors*

Description

Used to specify Normal priors for bayesGAM models

Usage

```
normal(param_values)
```

Arguments

param_values	Numeric vector of length 2 for the mean and standard deviation parameters

Details

For the beta and a parameters, the distribution is assumed to be unconstrained. For eps and lambda, the priors are half-normal with a support of strictly positive numbers.

References

Stan Development Team. 2018. Stan Modeling Language Users Guide and Reference Manual, Version 2.18.0

Examples

```
require(stats); require(graphics)
normal(c(0, 10))
```

np *Creates design matrices for univariate and bivariate applications*

Description

np accepts one or two numeric vectors of equal length as inputs. From these inputs, univariate or bivariate smoothing design matrices are produced. Currently available basis functions are truncated polynomials and thin plate splines. When bivariate smoothing is selected, np calls create_bivariate_design.

Usage

```
np(x1, x2 = NULL, num_knots = NULL, knots = NULL, basis = "tps", degree = 3)
```

Arguments

x1	numeric vector
x2	optional vector for bivariate non-parametric function
num_knots	optional number of knots
knots	optional numeric vector of knots
basis	character vector for basis function. tps for thin-plate spline and trunc.poly for truncated polynomial
degree	for truncated polynomial basis function

Value

list with the following elements:

- X parametric design matrix
- Z non-parametric design matrix
- knots numeric vector of knots for the model
- Xnms names of parameters passed to np
- basis selected basis function
- degree degree for truncated polynomial basis function

References

Ruppert, David, Matt P. Wand, and Raymond J. Carroll. *Semiparametric Regression*. No. 12. Cambridge university press, 2003. Section 5.6.

Matt Wand (2018). SemiPar: Semiparametric Regression. R package version 1.0-4.2.

Examples

```
x1 <- rnorm(100)
res <- np(x1, num_knots=10, basis="trunc.poly", degree=2)
res
```

---

plot *Additional plotting for MCMC visualization and diagnostics.*

Description

Marginal response smooth plot functions for parametric and nonparametric associations.

Usage

```
## S4 method for signature 'bayesGAMfit,missing'
plot(x, y, applylink = TRUE, ...)

## S4 method for signature 'predictPlotObject,missing'
plot(x, y, ...)

## S4 method for signature 'posteriorPredictObject,missing'
plot(x, y, ...)
```

Arguments

x	an object of class hmclearn, usually a result of a call to mh or hmc
y	unused
applylink	logical to indicate whether the inverse link function should be applied to the plots
...	optional additional arguments to pass to the ggplot2

Value

A list of *univariate* and *bivariate* plots generated by plot functions based on ggplot2

References

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

See Also

mcmc_plots

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=500, chains = 1)
plot(f)
```

---

posterior_predict *Posterior predictive samples from models fit by* bayesGAM

Description

Draw from the posterior predictive distribution

Usage

```
posterior_predict(object, ...)

## S4 method for signature 'bayesGAMfit'
posterior_predict(object, draws = NULL, ...)

## S4 method for signature 'glmModel'
posterior_predict(object, draws = NULL, results = NULL, ...)
```

Arguments

object	Object of type bayesGAMfit generated from bayesGAM.
...	Additional arguments for postrior_predict
draws	An integer indicating the number of draws to return. The default and maximum number of draws is the size of the posterior sample.
results	Matrix of HMC posterior samples

Value

a list of *D* by *N* matrices, where *D* is the number of draws from the posterior predictive distribution and *N* is the number of data points being predicted per draw.

References

Goodrich B, Gabry J, Ali I & Brilleman S. (2020). rstanarm: Bayesian applied regression modeling via Stan. R package version 2.19.3 https://mc-stan.org/rstanarm.

Jonah Gabry, Ben Goodrich and Martin Lysy (2020). rstantools: Tools for Developing R Packages Interfacing with 'Stan'. https://mc-stan.org/rstantools/, https://discourse.mc-stan.org/.

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=1000, chains = 1)
res <- posterior_predict(f, draws=100)
```

---

ppc_plots *Plotting for MCMC visualization and diagnostics provided by* bayesplot *package*

Description

Plots of Rhat statistics, ratios of effective sample size to total sample size, and autocorrelation of MCMC draws.

Usage

```
ppc_dens(object, ...)

## S4 method for signature 'bayesGAMfit'
ppc_dens(object, draws = NULL, ...)

## S4 method for signature 'posteriorPredictObject'
ppc_dens(object, ...)

ppc_dens_overlay(object, ...)

## S4 method for signature 'bayesGAMfit'
ppc_dens_overlay(object, draws = NULL, ...)

## S4 method for signature 'posteriorPredictObject'
ppc_dens_overlay(object, ...)

ppc_hist(object, ...)

## S4 method for signature 'bayesGAMfit'
ppc_hist(object, draws = NULL, ...)

## S4 method for signature 'posteriorPredictObject'
ppc_hist(object, ...)

ppc_boxplot(object, ...)

## S4 method for signature 'bayesGAMfit'
ppc_boxplot(object, draws = NULL, ...)

## S4 method for signature 'posteriorPredictObject'
ppc_boxplot(object, ...)

ppc_freqpoly(object, ...)

## S4 method for signature 'bayesGAMfit'
ppc_freqpoly(object, draws = NULL, ...)

## S4 method for signature 'posteriorPredictObject'
ppc_freqpoly(object, ...)

ppc_ecdf_overlay(object, ...)

## S4 method for signature 'bayesGAMfit'
ppc_ecdf_overlay(object, draws = NULL, ...)

## S4 method for signature 'posteriorPredictObject'
ppc_ecdf_overlay(object, ...)
```

Arguments

object	an object of class bayesGAMfit
...	optional additional arguments to pass to the bayesplot functions
draws	An integer indicating the number of draws to return. The default and maximum number of draws is the size of the posterior sample.

Value

These functions call various plotting functions from the bayesplot package, which returns a list including ggplot2 objects.

Plot Descriptions from the bayesplot package documentation

- ppc_hist(object, draws=NULL, ...) A separate histogram estimate is displayed for y and each dataset (row) in yrep. For these plots yrep should therefore contain only a small number of rows.
- ppc_boxplot(object, draws=NULL, ...) A separate box and whiskers plot is displayed for y and each dataset (row) in yrep. For these plots yrep should therefore contain only a small number of rows.
- ppc_freqpoly(object, draws=NULL, ...) A separate shaded frequency polygon is displayed for y and each dataset (row) in yrep. For these plots yrep should therefore contain only a small number of rows.
- ppc_dens(object, draws=NULL, ...) A separate smoothed kernel density estimate is displayed for y and each dataset (row) in yrep. For these plots yrep should therefore contain only a small number of rows.
- ppc_dens_overlay(object, draws=NULL, ...) Kernel density estimates of each dataset (row) in yrep are overlaid, with the distribution of y itself on top (and in a darker shade).
- ppc_ecdf_overlay(object, draws=NULL, ...) Empirical CDF estimates of each dataset (row) in yrep are overlaid, with the distribution of y itself on top (and in a darker shade).

References

Gabry, Jonah and Mahr, Tristan (2019). *bayesplot: Plotting for Bayesian Models*. https://mc-stan.org/bayesplot/

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., and Gelman, A (2019). *Visualization in Bayesian Workflow*. Journal of the Royal Statistical Society: Series A. Vol 182. Issue 2. p.389-402.

Gelman, A. and Rubin, D. (1992) *Inference from Iterative Simulation Using Multiple Sequences*. Statistical Science 7(4) 457-472.

Gelman, A., et. al. (2013) *Bayesian Data Analysis*. Chapman and Hall/CRC.

Gabry, J. , Simpson, D. , Vehtari, A. , Betancourt, M. and Gelman, A. (2019), Visualization in Bayesian workflow. J. R. Stat. Soc. A, 182: 389-402. doi:10.1111/rssa.12378.

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=500, chains = 1)
ppc_dens(f, draws=2)
```

`predict` *Posterior predictive samples from models fit by* `bayesGAM`*, but with new data*

Description

Draw from the posterior predictive distribution applied to new data

Usage

```
## S4 method for signature 'bayesGAMfit'
predict(object, newdata, draws = NULL, ...)
```

Arguments

`object`	Object of type `bayesGAMfit` generated from `bayesGAM`.
`newdata`	A data frame with new data applied to the `bayesGAMfit` object
`draws`	An integer indicating the number of draws to return. The default and maximum number of draws is the size of the posterior sample.
`...`	Additional arguments for `postrior_predict`

Value

a list of D by N matrices, where D is the number of draws from the posterior predictive distribution and N is the number of data points being predicted per draw.

References

Goodrich B, Gabry J, Ali I & Brilleman S. (2020). rstanarm: Bayesian applied regression modeling via Stan. R package version 2.19.3 https://mc-stan.org/rstanarm.

Jonah Gabry, Ben Goodrich and Martin Lysy (2020). rstantools: Tools for Developing R Packages Interfacing with 'Stan'. https://mc-stan.org/rstantools/, https://discourse.mc-stan.org/.

Examples

```
set.seed(432)
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=500, chains = 1)
newheights <- with(women, rnorm(10, mean = mean(height)), sd=sd(height))
women2 <- data.frame(height=newheights)

pred <- predict(f, women2, draws=100)
```

`reef` *Coral reef data from survey data on 6 sites*

Description

Data from 68 subjects

Usage

```
reef
```

Format

A data frame with 269 rows and 14 variables:

ZONE Management zone

site Name of the habitat site

complexity habitat benthic complexity

rugosity a measurement related to terrain complexity

LC cover of low complexity

HC cover of high complexity

SCORE1 PCA score 1 from Wilson, Graham, Polunin

SCORE2 PCA score 2 from Wilson, Graham, Polunin

macro indicator of race white

species fish species

abundance fish abundance

biomass fish biomass

Source

Data from supplementary material provided for Fisher, R., Wilson, S. K., Sin, T. M., Lee, A. C., and Langlois, T. J. (2018). *A simple function for full-subsets multiple regression in ecology with R*. Ecology and evolution, 8(12), 6104-6113.

References

Wilson, S. K., Graham, N. A. J., and Polunin, N. V. (2007). *Appraisal of visual assessments of habitat complexity and benthic composition on coral reefs*. Marine Biology, 151(3), 1069-1076.

showPrior *Display the priors used in* bayesGAM

Description

Prints a list of priors for β, λ, ϵ, and a, where applicable.

Usage

```
showPrior(object, ...)

## S4 method for signature 'bayesGAMfit'
showPrior(object, params = "all")
```

Arguments

object	Object of type bayesGAMfit generated from bayesGAM
...	Additional arguments for showPrior
params	character vector of the names of parameters to return • β beta • ϵ eps • λ lambda • $a]a$

Value

none

Examples

```
require(stats); require(graphics)
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter = 500, chains = 1)
showPrior(f)
```

st *Constructor function for Student-t priors*

Description

Used to specify student-t priors for bayesGAM models

Usage

```
st(param_values)
```

Arguments

param_values	Numeric vector of length 3 for the degrees of freedom, location, and scale parameter.

Details

For the beta and a parameters, the distribution is assumed to be unconstrained. For eps and lambda, the priors are half-normal with a support of strictly positive numbers.

References

Stan Development Team. 2018. Stan Modeling Language Users Guide and Reference Manual, Version 2.18.0

Examples

```
require(stats); require(graphics)
st(c(3,0,1))
```

---

summary *Summarizing Model Fits from* bayesGAM

Description

summary method for class bayesGAMfit

Usage

```
## S4 method for signature 'bayesGAMfit'
summary(object)
```

Arguments

object	an object of class hmclearn, usually a result of a call to mh or hmc

Value

Returns a matrix with posterior quantiles and the posterior scale reduction factor statistic for each parameter.

References

Stan Development Team (2020). RStan: the R interface to Stan. R package version 2.21.1.

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=500, chains = 1)

summary(f)
```

waic_bgam *Calls the* loo *package to calculate the widely applicable information criterion (WAIC)*

Description

Computes WAIC by calling the appropriate function from the loo package

Usage

```
waic_bgam(object, ...)

## S4 method for signature 'bayesGAMfit'
waic_bgam(object, ...)

## S4 method for signature 'array'
waic_bgam(object, ...)
```

Arguments

object	Object of type bayesGAMfit generated from bayesGAM.
...	Additional parameters to pass to pass to loo::waic

Value

a named list of class c("waic", "loo")

estimates A matrix with two columns ("Estimate", "SE") and three rows ("elpd_waic", "p_waic", "waic"). This contains point estimates and standard errors of the expected log pointwise predictive density (elpd_waic), the effective number of parameters (p_waic) and the information criterion waic (which is just -2 * elpd_waic, i.e., converted to deviance scale).

pointwise A matrix with three columns (and number of rows equal to the number of observations) containing the pointwise contributions of each of the above measures (elpd_waic, p_waic, waic).

References

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely application information criterion in singular learning theory. Journal of Machine Learning Research 11, 3571-3594.

Vehtari, A., Gelman, A., and Gabry, J. (2017a). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. Statistics and Computing. 27(5), 1413–1432. doi:10.1007/s11222-016-9696-4 (journal version, preprint arXiv:1507.04544).

Vehtari, A., Gelman, A., and Gabry, J. (2017b). Pareto smoothed importance sampling. preprint arXiv:1507.02646

Vehtari A, Gabry J, Magnusson M, Yao Y, Gelman A (2019). "loo: Efficient leave-one-out cross-validation and WAIC for Bayesian models." R package version 2.2.0, <URL: https://mc-stan.org/loo>.

Examples

```
f <- bayesGAM(weight ~ np(height), data = women,
              family = gaussian, iter=500, chains = 1)
waic_bgam(f)
```

Index