

# Package ‘bayesGAM’

March 17, 2022

**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](https://arxiv.org/abs/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](mailto: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, corrrplot, ggplot2, graphics, gridExtra, loo, methods, mlbench, Rcpp ( $\geq 0.12.0$ ), RcppParallel ( $\geq 5.0.1$ ), rstan ( $\geq 2.18.1$ ), rstantools ( $\geq 2.1.0.9000$ ), SemiPar, stats, geometry, MASS

**LinkingTo** BH ( $\geq 1.66.0$ ), Rcpp ( $\geq 0.12.0$ ), RcppEigen ( $\geq 0.3.3.3.0$ ), RcppParallel ( $\geq 5.0.1$ ), rstan ( $\geq 2.18.1$ ), StanHeaders ( $\geq 2.18.0$ )

**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

## R topics documented:

bayesGAM-package	3
bayesGAM	3
bayesGAMfit-class	5
bloodpressure	6
coefficients	7
create_bivariate_design	8
extract_log_lik_bgam	9
fitted	10
getDesign	10
getModelSlots	11
getSamples	12
getStanResults	13
L	14
loo_bgam	15
loo_compare_bgam	16
mcmc_plots	17
mvcorrplot	23
normal	24
np	24
plot	25
posterior_predict	26
ppc_plots	27
predict	30
reef	31
showPrior	32
st	32
summary	33
waic_bgam	34

---

bayesGAM-package	<i>The 'bayesGAM' package.</i>
------------------	--------------------------------

---

## 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	<i>bayesGAM fits a variety of regression models using Hamiltonian Monte Carlo</i>
----------	-----------------------------------------------------------------------------------

---

## Description

Based on [glm](#). bayesGAM is used to fit a variety of statistical models, including linear models, generalized linear 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 <a href="#">formula</a> object describing the model to be fitted.
random	(optional) specify a random intercept in the form ' <code>~var</code> '
family	distribution and link function for the model
data	(optional) data frame containing the variables in the model.
offset	Same as <a href="#">glm</a>
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 selected 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 <code>rstan::sampling</code> (e.g. <code>iter</code> , <code>chains</code> ).

**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	<i>Contains results from rstan as well as the design matrices and other data for the model.</i>
-------------------	-------------------------------------------------------------------------------------------------

---

## 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  $\beta$ ,  $\epsilon$ ,  $\lambda$ ,  $u$ ,  $\sigma$ ,  $a$ .

### 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 <ul style="list-style-type: none"> <li>• <math>\beta</math> beta</li> <li>• <math>\epsilon</math> eps</li> <li>• <math>\lambda</math> lambda</li> <li>• <math>a</math>]a</li> </ul>
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

<code>X1</code>	numeric vector for first variable
<code>X2</code>	numeric vector for second variable
<code>num_knots</code>	optional: number of knots
<code>knots</code>	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

<code>object</code>	Object of type <code>bayesGAMfit</code> generated from <code>bayesGAM</code> .
<code>...</code>	Additional parameters to pass to <code>loo::extract_log_lik</code>

## 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	<i>Extract fitted values from a model fit by bayesGAM</i>
--------	-----------------------------------------------------------

---

**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	<i>Design matrices from a bayesGAMfit object</i>
-----------	--------------------------------------------------

---

**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	<i>Return one or slots from the Stan model in bayesGAM</i>
---------------	------------------------------------------------------------

---

**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 <ul style="list-style-type: none"> <li>• X Fixed effects design matrix</li> <li>• Z Random effects design matrix</li> <li>• Zlst list of individual random effects design matrices that, combined, form Z</li> <li>• Zarray array of individual random effects design matrices. Used for multiple response models</li> <li>• max_col maximum number of columns of an individual Z matrix. Padding for STAN</li> <li>• y numeric response matrix</li> </ul>

- 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	<i>Returns the stanfit object generated by <b>rstan</b></i>
----------------	-------------------------------------------------------------

---

**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	<i>Calls the loo package to perform efficient approximate leave-one-out cross-validation on models fit with bayesGAM</i>
----------	--------------------------------------------------------------------------------------------------------------------------

---

**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  $-2 * \text{elpd\_loo}$ , 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)
```

---

<code>loo_compare_bgam</code>	<i>Calls the loo package to compare models fit by bayesGAMfit</i>
-------------------------------	-------------------------------------------------------------------

---

## 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

<code>object</code>	an object of class <code>bayesGAMfit</code>
<code>...</code>	optional additional arguments to pass to the <code>bayesplot</code> functions
<code>regex_pars</code>	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)
```

---

`mvsrplot`*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'  
mvsrplot(object, ...)
```

**Arguments**

<code>object</code>	model object of class <code>bayesGAMfit</code>
<code>...</code>	Additional parameters passed to <code>corrplot.mixed</code>

**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)  
mvsrplot(f)
```

---

normal	<i>Constructor function for Normal priors</i>
--------	-----------------------------------------------

---

**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	<i>Creates design matrices for univariate and bivariate applications</i>
----	--------------------------------------------------------------------------

---

**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	<i>Posterior predictive samples from models fit by bayesGAM</i>
-------------------	-----------------------------------------------------------------

---

**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 posterior_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	<i>Plotting for MCMC visualization and diagnostics provided by bayesplot package</i>
-----------	--------------------------------------------------------------------------------------

---

**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	<i>Posterior predictive samples from models fit by bayesGAM, but with new data</i>
---------	------------------------------------------------------------------------------------

---

**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 posterior_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	<i>Display the priors used in bayesGAM</i>
-----------	--------------------------------------------

---

**Description**

Prints a list of priors for  $\beta$ ,  $\lambda$ ,  $\epsilon$ , 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 <ul style="list-style-type: none"> <li>• <math>\beta</math> beta</li> <li>• <math>\epsilon</math> eps</li> <li>• <math>\lambda</math> lambda</li> <li>• <math>a</math>]a</li> </ul>

**Value**

none

**Examples**

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

---

st	<i>Constructor function for Student-t priors</i>
----	--------------------------------------------------

---

**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	<i>Calls the loo package to calculate the widely applicable information criterion (WAIC)</i>
-----------	----------------------------------------------------------------------------------------------

---

### 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 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  $-2 * \text{elpd\_waic}$ , 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

- \* **datasets**
  - bloodpressure, 6
  - reef, 31
- bayesGAM, 3
- bayesGAM-package, 3
- bayesGAMfit (bayesGAMfit-class), 5
- bayesGAMfit-class, 5
- bloodpressure, 6
- coef, bayesGAMfit-method (coefficients), 7
- coefficients, 7
- coefficients, bayesGAMfit-method (coefficients), 7
- create\_bivariate\_design, 8, 24
- extract\_log\_lik\_bgam, 9
- extract\_log\_lik\_bgam, bayesGAMfit-method (extract\_log\_lik\_bgam), 9
- family, 4
- fitted, 10
- fitted, bayesGAMfit-method (fitted), 10
- formula, 4
- getDesign, 10
- getDesign, bayesGAMfit-method (getDesign), 10
- getDesign, glmModel-method (getDesign), 10
- getModelSlots, 11
- getModelSlots, bayesGAMfit-method (getModelSlots), 11
- getSamples, 12
- getSamples, bayesGAMfit-method (getSamples), 12
- getSamples, glmModel-method (getSamples), 12
- getSamples, stanfit-method (getSamples), 12
- getStanResults, 13
- getStanResults, bayesGAMfit-method (getStanResults), 13
- glm, 3, 4
- L, 14
- loo\_bgam, 15
- loo\_bgam, array-method (loo\_bgam), 15
- loo\_bgam, bayesGAMfit-method (loo\_bgam), 15
- loo\_compare\_bgam, 16
- loo\_compare\_bgam, bayesGAMfit-method (loo\_compare\_bgam), 16
- mcmc\_acf (mcmc\_plots), 17
- mcmc\_acf, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_acf\_bar (mcmc\_plots), 17
- mcmc\_acf\_bar, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_areas (mcmc\_plots), 17
- mcmc\_areas, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_dens (mcmc\_plots), 17
- mcmc\_dens, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_hex (mcmc\_plots), 17
- mcmc\_hex, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_hist (mcmc\_plots), 17
- mcmc\_hist, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_hist\_by\_chain (mcmc\_plots), 17
- mcmc\_hist\_by\_chain, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_intervals (mcmc\_plots), 17
- mcmc\_intervals, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_neff (mcmc\_plots), 17

- mcmc\_neff, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_neff\_data (mcmc\_plots), 17
- mcmc\_neff\_data, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_neff\_hist (mcmc\_plots), 17
- mcmc\_neff\_hist, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_pairs (mcmc\_plots), 17
- mcmc\_pairs, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_plots, 17, 26
- mcmc\_rhat (mcmc\_plots), 17
- mcmc\_rhat, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_rhat\_data (mcmc\_plots), 17
- mcmc\_rhat\_data, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_rhat\_hist (mcmc\_plots), 17
- mcmc\_rhat\_hist, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_scatter (mcmc\_plots), 17
- mcmc\_scatter, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_trace (mcmc\_plots), 17
- mcmc\_trace, bayesGAMfit-method (mcmc\_plots), 17
- mcmc\_violin (mcmc\_plots), 17
- mcmc\_violin, bayesGAMfit-method (mcmc\_plots), 17
- mvcorrplot, 23
- mvcorrplot, bayesGAMfit-method (mvcorrplot), 23
  
- normal, 24
- np, 24
  
- pareto-k-diagnostic, 16
- plot, 25
- plot, bayesGAMfit, missing-method (plot), 25
- plot, posteriorPredictObject, missing-method (plot), 25
- plot, predictPlotObject, missing-method (plot), 25
- posterior\_predict, 26
- posterior\_predict, bayesGAMfit-method (posterior\_predict), 26
- posterior\_predict, glmModel-method (posterior\_predict), 26
- ppc\_boxplot (ppc\_plots), 27
- ppc\_boxplot, bayesGAMfit-method (ppc\_plots), 27
- ppc\_boxplot, posteriorPredictObject-method (ppc\_plots), 27
- ppc\_dens (ppc\_plots), 27
- ppc\_dens, bayesGAMfit-method (ppc\_plots), 27
- ppc\_dens, posteriorPredictObject-method (ppc\_plots), 27
- ppc\_dens\_overlay (ppc\_plots), 27
- ppc\_dens\_overlay, bayesGAMfit-method (ppc\_plots), 27
- ppc\_dens\_overlay, posteriorPredictObject-method (ppc\_plots), 27
- ppc\_ecdf\_overlay (ppc\_plots), 27
- ppc\_ecdf\_overlay, bayesGAMfit-method (ppc\_plots), 27
- ppc\_ecdf\_overlay, posteriorPredictObject-method (ppc\_plots), 27
- ppc\_freqpoly (ppc\_plots), 27
- ppc\_freqpoly, bayesGAMfit-method (ppc\_plots), 27
- ppc\_freqpoly, posteriorPredictObject-method (ppc\_plots), 27
- ppc\_hist (ppc\_plots), 27
- ppc\_hist, bayesGAMfit-method (ppc\_plots), 27
- ppc\_hist, posteriorPredictObject-method (ppc\_plots), 27
- ppc\_plots, 27
- predict, 30
- predict, bayesGAMfit-method (predict), 30
- psis(), 16
  
- reef, 31
- show, bayesGAMfit-method (bayesGAMfit-class), 5
- showPrior, 32
- showPrior, bayesGAMfit-method (showPrior), 32
- st, 32
- summary, 33
- summary, bayesGAMfit-method (summary), 33
- waic\_bgam, 34

`waic_bgam`, array-method (`waic_bgam`), [34](#)  
`waic_bgam`, bayesGAMfit-method  
(`waic_bgam`), [34](#)