Package 'hmclearn'

October 13, 2022

Type Package

Title Fit Statistical Models Using Hamiltonian Monte Carlo

Version 0.0.5

Maintainer Samuel Thomas <samthoma@iu.edu>

Description Provide users with a framework to learn the intricacies of the Hamiltonian Monte Carlo algorithm with hands-on experience by tuning and fitting their own models. All of the code is written in R. Theoretical references are listed below:.
Neal, Radford (2011) ``Handbook of Markov Chain Monte Carlo" ISBN: 978-1420079418,
Betancourt, Michael (2017) ``A Conceptual Introduction to Hamiltonian Monte Carlo" <arXiv:1701.02434>,
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.

Depends R (>= 3.6)

License GPL-3

Encoding UTF-8

Language en-US

LazyData true

Suggests knitr, rmarkdown, Matrix, lme4, carData, mlbench, ggplot2, mlmRev, testthat, MCMCpack

RoxygenNote 7.1.1

Imports bayesplot, parallel, MASS, mvtnorm

VignetteBuilder knitr

NeedsCompilation no

Author Samuel Thomas [cre, aut], Wanzhu Tu [ctb]

Repository CRAN

Date/Publication 2020-10-05 06:40:02 UTC

R topics documented:

`coef.hmclearn` *Extract Model Coefficients*

Description

Method for `hmclearn` objects created by `mh` and `hmc` functions. Extracts the specified quantile of the posterior.

Usage

```
## S3 method for class 'hmclearn'
coef(object, burnin = NULL, prob = 0.5, ...)
```

Arguments

`object`	an object of class `hmclearn`, usually a result of a call to `mh` or `hmc`
`burnin`	optional numeric parameter for the number of initial MCMC samples to omit from the summary
`prob`	quantile to extract coefficients
`...`	additional arguments to pass to `quantile`

Value

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

Examples

```
# Linear regression example
set.seed(521)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

f1 <- hmc(N = 500,
          theta.init = c(rep(0, 4), 1),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = linear_posterior,
          glogPOSTERIOR = g_linear_posterior,
          varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
          param=list(y=y, X=X), parallel=FALSE, chains=1)

summary(f1)
coef(f1)
```

---

diagplots *Diagnostic plots for* hmclearn

Description

Plots histograms of the posterior estimates. Optionally, displays the 'actual' values given a simulated dataset.

Usage

```
diagplots(
  object,
  burnin = NULL,
  plotfun = 2,
  comparison.theta = NULL,
  cols = NULL,
  ...
)
```

Arguments

object	an object of class hmclearn, usually a result of a call to mh or hmc
burnin	optional numeric parameter for the number of initial MCMC samples to omit from the summary

plotfun	integer 1 or 2 indicating which plots to display. 1 shows trace plots. 2 shows a histogram
comparison.theta	optional numeric vector of true parameter values
cols	optional integer index indicating which parameters to display
...	currently unused

Value

Returns a customized ggplot object

Examples

```
# Linear regression example
set.seed(522)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

f <- hmc(N = 1000,
          theta.init = c(rep(0, 4), 1),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = linear_posterior,
          glogPOSTERIOR = g_linear_posterior,
          varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
          param=list(y=y, X=X), parallel=FALSE, chains=1)

diagplots(f, burnin=300, comparison.theta=c(betavals, 2*log(.2)))
```

---

diagplots.hmclearn *Diagnostic plots for* hmclearn

Description

Plots histograms of the posterior estimates. Optionally, displays the 'actual' values given a simulated dataset.

Usage

```
## S3 method for class 'hmclearn'
diagplots(
  object,
  burnin = NULL,
  plotfun = 2,
  comparison.theta = NULL,
  cols = NULL,
  ...
)
```

Arguments

object	an object of class hmclearn, usually a result of a call to mh or hmc
burnin	optional numeric parameter for the number of initial MCMC samples to omit from the summary
plotfun	integer 1 or 2 indicating which plots to display. 1 shows trace plots. 2 shows a histogram
comparison.theta	optional numeric vector of parameter values to compare to the Bayesian estimates
cols	optional integer index indicating which parameters to display
...	currently unused

Value

Returns a customized ggplot object

Examples

```
# Linear regression example
set.seed(522)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

f <- hmc(N = 1000,
          theta.init = c(rep(0, 4), 1),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = linear_posterior,
          glogPOSTERIOR = g_linear_posterior,
          varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
          param=list(y=y, X=X), parallel=FALSE, chains=1)

diagplots(f, burnin=300, comparison.theta=c(betavals, 2*log(.2)))
```

Drugs *Student Drug Usage Dataset*

Description

Data from a survey of 2276 high school students about drug usage

Usage

```
Drugs
```

Format

A data frame with 8 rows and 4 variables:

A Alcohol usage (Yes/No)

C Cigarette usage (Yes/No)

M Marijuana usage (Yes/No)

count number of responses

Source

Data originally provided by Harry Khamis, Wright State University

References

Agresti, A. (2015). *Foundations of linear and generalized linear models*. John Wiley & Sons. http://users.stat.ufl.edu/~aa/glm/data/Drugs.dat

---

`Endometrial` *Endometrial Cancer Dataset*

Description

Data from a study about Endometrial Cancer

Usage

```
Endometrial
```

Format

A data frame with 79 rows and 4 variables:

NV Neovasculation risk factor indicator (0=Absent, 1=Present)

PI Pulsatility index of arteria uterina

EH Endometrium height

HG histology of patient (0=Low, 1=High)

Source

Heinze, G., & Schemper, M. (2002). *A solution to the problem of separation in logistic regression*. Statistics in medicine, 21(16), 2409-2419.

References

Agresti, A. (2015). *Foundations of linear and generalized linear models*. John Wiley & Sons. http://users.stat.ufl.edu/~aa/glm/data/Endometrial.dat

Gdat *Count of Fresh Gopher Tortoise Shells*

Description

A dataset containing the count of fresh gopher shells by area.

Usage

```
Gdat
```

Format

A data frame with 30 rows and 7 variables:

Site name of site

year years 2004, 2005, 2006

shells count of shells

type fresh water

Area area of the site

density estimated tortoise density

prev Seroprevalence to Mycoplasma agassizii

Source

Ozgul, A., Oli, M. K., Bolker, B. M., & Perez-Heydrich, C. (2009). *Upper respiratory tract disease, force of infection, and effects on survival of gopher tortoises*. Ecological Applications, 19(3), 786–798

References

Fox, G. A., Negrete-Yankelevich, S., & Sosa, V. J. (Eds.). (2015). *Ecological statistics: contemporary theory and application*. Oxford University Press, USA.

Bolker, Ben (2018) GLMM Worked Examples https://bbolker.github.io/mixedmodels-misc/ecostats_chap.html

hmc *Fit a generic model using Hamiltonian Monte Carlo (HMC)*

Description

This function runs the HMC algorithm on a generic model provided the `logPOSTERIOR` and gradient `glogPOSTERIOR` functions. All parameters specified within the list `param`are passed to these two functions. The tuning parameters `epsilon` and `L` are passed to the Leapfrog algorithm.

Usage

```
hmc(
  N = 10000,
  theta.init,
  epsilon = 0.01,
  L = 10,
  logPOSTERIOR,
  glogPOSTERIOR,
  randlength = FALSE,
  Mdiag = NULL,
  constrain = NULL,
  verbose = FALSE,
  varnames = NULL,
  param = list(),
  chains = 1,
  parallel = FALSE,
  ...
)
```

Arguments

`N`	Number of MCMC samples
`theta.init`	Vector of initial values for the parameters
`epsilon`	Step-size parameter for `leapfrog`
`L`	Number of `leapfrog` steps parameter
`logPOSTERIOR`	Function to calculate and return the log posterior given a vector of values of `theta`
`glogPOSTERIOR`	Function to calculate and return the gradient of the log posterior given a vector of values of `theta`
`randlength`	Logical to determine whether to apply some randomness to the number of leapfrog steps tuning parameter `L`
`Mdiag`	Optional vector of the diagonal of the mass matrix `M`. Defaults to unit diagonal.
`constrain`	Optional vector of which parameters in `theta` accept positive values only. Default is that all parameters accept all real numbers

verbose	Logical to determine whether to display the progress of the HMC algorithm
varnames	Optional vector of theta parameter names
param	List of additional parameters for logPOSTERIOR and glogPOSTERIOR
chains	Number of MCMC chains to run
parallel	Logical to set whether multiple MCMC chains should be run in parallel
...	Additional parameters for logPOSTERIOR

Value

Object of class hmclearn

Elements for hmclearn objects

N Number of MCMC samples

theta Nested list of length N of the sampled values of theta for each chain

thetaCombined List of dataframes containing sampled values, one for each chain

r List of length N of the sampled momenta

theta.all Nested list of all parameter values of theta sampled prior to accept/reject step for each

r.all List of all values of the momenta r sampled prior to accept/reject

accept Number of accepted proposals. The ratio accept / N is the acceptance rate

accept_v Vector of length N indicating which samples were accepted

M Mass matrix used in the HMC algorithm

algorithm HMC for Hamiltonian Monte Carlo

varnames Optional vector of parameter names

chains Number of MCMC chains

Available logPOSTERIOR and glogPOSTERIOR functions

linear_posterior Linear regression: log posterior

g_linear_posterior Linear regression: gradient of the log posterior

logistic_posterior Logistic regression: log posterior

g_logistic_posterior Logistic regression: gradient of the log posterior

poisson_posterior Poisson (count) regression: log posterior

g_poisson_posterior Poisson (count) regression: gradient of the log posterior

lmm_posterior Linear mixed effects model: log posterior

g_lmm_posterior Linear mixed effects model: gradient of the log posterior

glmm_bin_posterior Logistic mixed effects model: log posterior

g_glmm_bin_posterior Logistic mixed effects model: gradient of the log posterior

glmm_poisson_posterior Poisson mixed effects model: log posterior

g_glmm_poisson_posterior Poisson mixed effects model: gradient of the log posterior

Author(s)

Samuel Thomas <samthoma@iu.edu>, Wanzhu Tu <wtu@iu.edu>

References

Neal, Radford. 2011. *MCMC Using Hamiltonian Dynamics.* In Handbook of Markov Chain Monte Carlo, edited by Steve Brooks, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng, 116–62. Chapman; Hall/CRC.

Betancourt, Michael. 2017. *A Conceptual Introduction to Hamiltonian Monte Carlo*.

Thomas, S., Tu, W. 2020. *Learning Hamiltonian Monte Carlo in R*.

Examples

```
# Linear regression example
set.seed(521)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

fm1_hmc <- hmc(N = 500,
          theta.init = c(rep(0, 4), 1),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = linear_posterior,
          glogPOSTERIOR = g_linear_posterior,
          varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
          param=list(y=y, X=X), parallel=FALSE, chains=1)

summary(fm1_hmc, burnin=100)


# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)

fm2_hmc <- hmc(N = 500,
          theta.init = rep(0, 3),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = poisson_posterior,
          glogPOSTERIOR = g_poisson_posterior,
          varnames = paste0("beta", 0:2),
          param = list(y=y, X=X),
          parallel=FALSE, chains=1)

summary(fm2_hmc, burnin=100)
```

`hmc.fit` *Fitter function for Hamiltonian Monte Carlo (HMC)*

Description

This is the basic computing function for HMC and should not be called directly except by experienced users.

Usage

```
hmc.fit(
  N,
  theta.init,
  epsilon,
  L,
  logPOSTERIOR,
  glogPOSTERIOR,
  varnames = NULL,
  randlength = FALSE,
  Mdiag = NULL,
  constrain = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

`N`	Number of MCMC samples
`theta.init`	Vector of initial values for the parameters
`epsilon`	Step-size parameter for `leapfrog`
`L`	Number of `leapfrog` steps parameter
`logPOSTERIOR`	Function to calculate and return the log posterior given a vector of values of `theta`
`glogPOSTERIOR`	Function to calculate and return the gradient of the log posterior given a vector of values of `theta`
`varnames`	Optional vector of theta parameter names
`randlength`	Logical to determine whether to apply some randomness to the number of leapfrog steps tuning parameter `L`
`Mdiag`	Optional vector of the diagonal of the mass matrix `M`. Defaults to unit diagonal.
`constrain`	Optional vector of which parameters in `theta` accept positive values only. Default is that all parameters accept all real numbers
`verbose`	Logical to determine whether to display the progress of the HMC algorithm
`...`	Additional parameters for `logPOSTERIOR` and `glogPOSTERIOR`

Value

List for hmc

Elements for hmclearn objects

N Number of MCMC samples

theta Nested list of length N of the sampled values of theta for each chain

thetaCombined List of dataframes containing sampled values, one for each chain

r List of length N of the sampled momenta

theta.all Nested list of all parameter values of theta sampled prior to accept/reject step for each

r.all List of all values of the momenta r sampled prior to accept/reject

accept Number of accepted proposals. The ratio accept / N is the acceptance rate

accept_v Vector of length N indicating which samples were accepted

M Mass matrix used in the HMC algorithm

algorithm HMC for Hamiltonian Monte Carlo

References

Neal, Radford. 2011. *MCMC Using Hamiltonian Dynamics.* In Handbook of Markov Chain Monte Carlo, edited by Steve Brooks, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng, 116–62. Chapman; Hall/CRC.

Betancourt, Michael. 2017. *A Conceptual Introduction to Hamiltonian Monte Carlo*.

Thomas, S., Tu, W. 2020. *Learning Hamiltonian Monte Carlo in R*.

Examples

```
# Logistic regression example
X <- cbind(1, seq(-100, 100, by=0.25))
betavals <- c(-0.9, 0.2)
lodds <- X %*% betavals
prob1 <- as.numeric(1 / (1 + exp(-lodds)))

set.seed(9874)
y <- sapply(prob1, function(xx) {
  sample(c(0, 1), 1, prob=c(1-xx, xx))
})

f1 <- hmc.fit(N = 500,
          theta.init = rep(0, 2),
          epsilon = c(0.1, 0.002),
          L = 10,
          logPOSTERIOR = logistic_posterior,
          glogPOSTERIOR = g_logistic_posterior,
          y=y, X=X)

f1$accept / f1$N
```

hmclearn-glm-posterior

Sample log posterior and gradient functions for select generalized linear models and mixed effect models

Description

These functions can be used to fit common generalized linear models and mixed effect models. See the accompanying vignettes for details on the derivations of the log posterior and gradient. In addition, these functions can be used as templates to build custom models to fit using HMC.

Usage

```
linear_posterior(theta, y, X, a = 1e-04, b = 1e-04, sig2beta = 1000)

g_linear_posterior(theta, y, X, a = 1e-04, b = 1e-04, sig2beta = 1000)

logistic_posterior(theta, y, X, sig2beta = 1000)

g_logistic_posterior(theta, y, X, sig2beta = 1000)

poisson_posterior(theta, y, X, sig2beta = 1000)

g_poisson_posterior(theta, y, X, sig2beta = 1000)

lmm_posterior(
  theta,
  y,
  X,
  Z,
  n,
  d,
  nrandom = 1,
  nugamma = 1,
  nuxi = 1,
  Agamma = 25,
  Axi = 25,
  sig2beta = 1000
)

g_lmm_posterior(
  theta,
  y,
  X,
  Z,
  n,
  d,
```

```
  nrandom = 1,
  nugamma = 1,
  nuxi = 1,
  Agamma = 25,
  Axi = 25,
  sig2beta = 1000
)

glmm_bin_posterior(
  theta,
  y,
  X,
  Z,
  n,
  nrandom = 1,
  nuxi = 1,
  Axi = 25,
  sig2beta = 1000
)

g_glmm_bin_posterior(
  theta,
  y,
  X,
  Z,
  n,
  nrandom = 1,
  nuxi = 1,
  Axi = 25,
  sig2beta = 1000
)

glmm_poisson_posterior(
  theta,
  y,
  X,
  Z,
  n,
  nrandom = 1,
  nuxi = 1,
  Axi = 25,
  sig2beta = 1000
)

g_glmm_poisson_posterior(
  theta,
  y,
  X,
```

```
  Z,
  n,
  nrandom = 1,
  nuxi = 1,
  Axi = 25,
  sig2beta = 1000
)
```

Arguments

theta	vector of parameters. See details below for the order of parameters for each model
y	numeric vector for the dependent variable for all models
X	numeric design matrix of fixed effect parameters for all models
a	hyperparameter for the Inverse Gamma shape parameter for σ_ϵ in linear regression models
b	hyperparameter for the Inverse Gamma scale parameter for σ_ϵ in linear regression models
sig2beta	diagonal covariance of prior for linear predictors is multivariate normal with mean 0 for linear regression and linear mixed effect models.
Z	numeric design matrix of random effect parameters for all mixed effects models
n	number of observations for standard glm models, or number of subjects for all mixed effect models
d	number of observations per subject for mixed effects models, but an input for linear mixed effect models only.
nrandom	number of random effects covariance parameters for all mixed effects models
nugamma	hyperparameter ν for the half-t prior of the log transformed error for linear mixed effects model γ
nuxi	hyperparameter ν for the half-t prior of the random effects diagonal for all mixed effects models ξ
Agamma	hyperparameter A for the half-t prior of the log transformed error for linear mixed effects model γ
Axi	hyperparameter A for the half-t prior of the random effects diagonal for all mixed effects modelsξ

Value

Numeric vector for the log posterior or gradient of the log posterior

Generalized Linear Models with available posterior and gradient functions

'linear_posterior(theta, y, X, a=1e-4, b=1e-4, sig2beta = 1e3)' The log posterior function for linear regression

$$f(y|X,\beta,\sigma) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)^T(y - X\beta)\right)$$

with priors $p(\sigma^2) \sim IG(a, b)$ and $\beta \sim N(0, \sigma^2_\beta I)$. The variance term is log transformed $\gamma = \log \sigma$ The input parameter vector $theta$ is of length k. The first $k-1$ parameters are for β, and the last parameter is γ Note that the Inverse Gamma prior can be problematic for certain applications with low variance, such as hierarchical models. See Gelman (2006)

'g_linear_posterior(theta, y, X, a = 1e-04, b = 1e-04, sig2beta=1e3)' Gradient of the log posterior for a linear regression model with Normal prior for the linear parameters and Inverse Gamma for the error term.

$$f(y|X, \beta, \sigma) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)^T(y - X\beta)\right)$$

with priors $p(\sigma^2) \sim IG(a, b)$ and $\beta \sim N(0, \sigma^2_\beta I)$. The variance term is log transformed $\gamma = \log \sigma$ The input parameter vector $theta$ is of length k. The first $k-1$ parameters are for β, and the last parameter is γ Note that the Inverse Gamma prior can be problematic for certain applications with low variance, such as hierarchical models. See Gelman (2006)

'logistic_posterior(theta, y, X, sig2beta=1e3) ' Log posterior for a logistic regression model with Normal prior for the linear parameters. The likelihood function for logistic regression

$$f(\beta|X, y) = \prod_{i=1}^{n} \left(\frac{1}{1 + e^{-X_i\beta}}\right)^{y_i} \left(\frac{e^{-X_i\beta}}{1 + e^{-X_i\beta}}\right)^{1-y_i}$$

with priors $\beta \sim N(0, \sigma^2_\beta I)$. The input parameter vector $theta$ is of length k, containing parameter values for β

'g_logistic_posterior(theta, y, X, sig2beta=1e3) ' Gradient of the log posterior for a logistic regression model with Normal prior for the linear parameters. The likelihood function for logistic regression

$$f(\beta|X, y) = \prod_{i=1}^{n} \left(\frac{1}{1 + e^{-X_i\beta}}\right)^{y_i} \left(\frac{e^{-X_i\beta}}{1 + e^{-X_i\beta}}\right)^{1-y_i}$$

with priors $\beta \sim N(0, \sigma^2_\beta I)$. The input parameter vector $theta$ is of length k, containing parameter values for β

'poisson_posterior(theta, y, X, sig2beta=1e3) ' Log posterior for a Poisson regression model with Normal prior for the linear parameters. The likelihood function for poisson regression

$$f(\beta|y, X) = \prod_{i=1}^{n} \frac{e^{-e^{X_i\beta}} e^{y_i X_i \beta}}{y_i!}$$

with priors $\beta \sim N(0, \sigma^2_\beta I)$. The input parameter vector $theta$ is of length k, containing parameter values for β

'g_poisson_posterior(theta, y, X, sig2beta=1e3) ' Gradient of the log posterior for a Poisson regression model with Normal prior for the linear parameters. The likelihood function for poisson regression

$$f(\beta|y, X) = \prod_{i=1}^{n} \frac{e^{-e^{X_i\beta}} e^{y_i X_i \beta}}{y_i!}$$

with priors $\beta \sim N(0, \sigma^2_\beta I)$. The input parameter vector $theta$ is of length k, containing parameter values for β

Generalized Linear Mixed Effect with available posterior and gradient functions

'lmm_posterior(theta, y, X, Z, n, d, nrandom = 1, nueps = 1, nuxi = 1, Aeps = 25, Axi = 25, sig2beta = 1e3) '
The log posterior function for linear mixed effects regression

$$f(y|\beta, u, \sigma_\epsilon) \propto (\sigma_\epsilon^2)^{-nd/2} e^{-\frac{1}{2\sigma_\epsilon^2}(y-X\beta-Zu)^T(y-X\beta-Zu)}$$

with priors $\beta \sim N(0, \sigma_\beta^2 I)$, $\sigma_\epsilon \sim half-t(A_\epsilon, nu_\epsilon)$, $\lambda \sim half-t$. The vector ξ is the diagonal of the covariance G log transformed hyperprior where $u \sim N(0, G$, $\xi = \log\lambda$ and A_ξ, ν_ξ are parameters for the transformed distribution The standard deviation of the error is log transformed, where $\gamma = \log\sigma_\epsilon$ and $\sigma_\epsilon \sim half-t$. The parameters for γ are A_γ, ν_γ The input parameter vector $theta$ is of length k. The order of parameters for the vector is β, τ, γ, ξ.

'g_lmm_posterior(theta, y, X, Z, n, d, nrandom = 1, nueps = 1, nuxi = 1, Aeps = 25, Axi = 25, sig2beta = 1e3)'
Gradient of the log posterior for a linear mixed effects regression model

$$f(y|\beta, u, \sigma_\epsilon) \propto (\sigma_\epsilon^2)^{-n/2} e^{-\frac{1}{2\sigma_\epsilon^2}(y-X\beta-Zu)^T(y-X\beta-Zu)}$$

with priors $\beta \sim N(0, \sigma_\beta^2 I)$, $\sigma_\epsilon \sim half-t(A_\epsilon, nu_\epsilon)$, $\lambda \sim half-t$. The vector ξ is the diagonal of the covariance G log transformed hyperprior where $u \sim N(0, G$, $\xi = \log\lambda$ and A_ξ, ν_ξ are parameters for the transformed distribution The standard deviation of the error is log transformed, where $\gamma = \log\sigma_\epsilon$ and $\sigma_\epsilon \sim half-t$. The parameters for γ are A_γ, ν_γ The input parameter vector $theta$ is of length k. The order of parameters for the vector is β, τ, γ, ξ

'glmm_bin_posterior(theta, y, X, Z, n, nrandom = 1, nuxi = 1, Axi = 25, sig2beta=1e3)' The log posterior function for logistic mixed effects regression

$$f(y|X, Z, \beta, u) = \prod_{i=1}^{n}\prod_{j=1}^{d}\left(\frac{1}{1+e^{-X_i\beta-Z_{ij}u_i}}\right)^{y_{ij}}\left(\frac{e^{-X_i\beta-Z_{ij}u_i}}{1+e^{-X_i\beta-Z_{ij}u_i}}\right)^{1-y_{ij}}$$

with priors $\beta \sim N(0, \sigma_\beta^2 I)$, $\sigma_\epsilon \sim half-t(A_\epsilon, nu_\epsilon)$, $\lambda \sim half-t(A_\lambda, nu_\lambda)$. The vector λ is the diagonal of the covariance G hyperprior where $u \sim N(0, G$, $\xi = \log\lambda$ and A_ξ, ν_ξ are parameters for the transformed distribution The input parameter vector $theta$ is of length k. The order of parameters for the vector is β, τ, ξ

'g_glmm_bin_posterior(theta, y, X, Z, n, nrandom = 1, nuxi = 1, Axi = 25, sig2beta = 1e3) ' Gradient of the log posterior function for logistic mixed effects regression

$$f(y|X, Z, \beta, u) = \prod_{i=1}^{n}\prod_{j=1}^{m}\left(\frac{1}{1+e^{-X_i\beta-Z_{ij}u_i}}\right)^{y_{ij}}\left(\frac{e^{-X_i\beta-Z_{ij}u_i}}{1+e^{-X_i\beta-Z_{ij}u_i}}\right)^{1-y_{ij}}$$

with priors $\beta \sim N(0, \sigma_\beta^2 I)$, $\sigma_\epsilon \sim half-t(A_\epsilon, nu_\epsilon)$, $\lambda \sim half-t(A_\lambda, nu_\lambda)$. The vector λ is the diagonal of the covariance G hyperprior where $u \sim N(0, G$, $\xi = \log\lambda$ and A_ξ, ν_ξ are parameters for the transformed distribution The input parameter vector $theta$ is of length k. The order of parameters for the vector is β, τ, ξ

'glmm_poisson_posterior(theta, y, X, Z, n, nrandom = 1, nuxi = 1, Axi = 25, sig2beta = 1e3) '
Log posterior for a Poisson mixed effect regression

$$f(y|X, Z, \beta, u) = \prod_{i=1}^{n}\prod_{j=1}^{m}\frac{e^{-e^{X_i\beta+Z_{ij}u_{ij}}}e^{y_i(X_i\beta+Z_{ij}u_{ij})}}{y_i!}$$

with priors $\beta \sim N(0, \sigma_\beta^2 I)$, $\sigma_\epsilon \sim half - t(A_\epsilon, nu_\epsilon)$, $\lambda \sim half - t(A_\lambda, nu_\lambda)$. The vector λ is the diagonal of the covariance G hyperprior where $u \sim N(0, G$, $\xi = \log \lambda$ and A_ξ, ν_ξ are parameters for the transformed distribution The input parameter vector $theta$ is of length k. The order of parameters for the vector is β, τ, ξ

'g_glmm_poisson_posterior(theta, y, X, Z, n, nrandom = 1, nuxi = 1, Axi = 25, sig2beta = 1e3) '
Gradient of the log posterior for a Poisson mixed effect regression

$$f(y|X, Z, \beta, u) = \prod_{i=1}^{n} \prod_{j=1}^{m} \frac{e^{-e^{X_i\beta + Z_{ij}u_{ij}}} e^{y_i(X_i\beta + Z_{ij}u_{ij})}}{y_i!}$$

with priors $\beta \sim N(0, \sigma_\beta^2 I)$, $\sigma_\epsilon \sim half - t(A_\epsilon, nu_\epsilon)$, $\lambda \sim half - t(A_\lambda, nu_\lambda)$. The vector λ is the diagonal of the covariance G hyperprior where $u \sim N(0, G$, $\xi = \log \lambda$ and A_ξ, ν_ξ are parameters for the transformed distribution The input parameter vector $theta$ is of length k. The order of parameters for the vector is β, τ, ξ

References

Gelman, A. (2006). *Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper)*. Bayesian analysis, 1(3), 515-534.

Chan, J. C. C., & Jeliazkov, I. (2009). *MCMC estimation of restricted covariance matrices*. Journal of Computational and Graphical Statistics, 18(2), 457-480.

Betancourt, M., & Girolami, M. (2015). *Hamiltonian Monte Carlo for hierarchical models*. Current trends in Bayesian methodology with applications, 79, 30.

Examples

```
# Linear regression example
set.seed(521)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

f1_hmc <- hmc(N = 500,
          theta.init = c(rep(0, 4), 1),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = linear_posterior,
          glogPOSTERIOR = g_linear_posterior,
          varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
          param=list(y=y, X=X), parallel=FALSE, chains=1)

summary(f1_hmc, burnin=100)


# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)
```

```
f2_hmc <- hmc(N = 500,
          theta.init = rep(0, 3),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = poisson_posterior,
          glogPOSTERIOR = g_poisson_posterior,
          varnames = paste0("beta", 0:2),
          param = list(y=y, X=X),
          parallel=FALSE, chains=1)
```

`hmclearn-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, ...)

## S3 method for class 'hmclearn'
mcmc_intervals(object, burnin = NULL, ...)

mcmc_areas(object, ...)

## S3 method for class 'hmclearn'
mcmc_areas(object, burnin = NULL, ...)

mcmc_hist(object, ...)

## S3 method for class 'hmclearn'
mcmc_hist(object, burnin = NULL, ...)

mcmc_hist_by_chain(object, ...)

## S3 method for class 'hmclearn'
mcmc_hist_by_chain(object, burnin = NULL, ...)

mcmc_dens(object, ...)

## S3 method for class 'hmclearn'
mcmc_dens(object, burnin = NULL, ...)
```

```
mcmc_scatter(object, ...)

## S3 method for class 'hmclearn'
mcmc_scatter(object, burnin = NULL, ...)

mcmc_hex(object, ...)

## S3 method for class 'hmclearn'
mcmc_hex(object, burnin = NULL, ...)

mcmc_pairs(object, ...)

## S3 method for class 'hmclearn'
mcmc_pairs(object, burnin = NULL, ...)

mcmc_acf(object, ...)

## S3 method for class 'hmclearn'
mcmc_acf(object, burnin = NULL, ...)

mcmc_acf_bar(object, ...)

## S3 method for class 'hmclearn'
mcmc_acf_bar(object, burnin = NULL, ...)

mcmc_trace(object, ...)

## S3 method for class 'hmclearn'
mcmc_trace(object, burnin = NULL, ...)

mcmc_rhat(object, ...)

## S3 method for class 'hmclearn'
mcmc_rhat(object, burnin = NULL, ...)

mcmc_rhat_hist(object, ...)

## S3 method for class 'hmclearn'
mcmc_rhat_hist(object, burnin = NULL, ...)

mcmc_neff(object, ...)

## S3 method for class 'hmclearn'
mcmc_neff(object, burnin = NULL, lagmax = NULL, ...)

mcmc_neff_hist(object, ...)

## S3 method for class 'hmclearn'
```

```
mcmc_neff_hist(object, burnin = NULL, lagmax = NULL, ...)

mcmc_neff_data(object, ...)

## S3 method for class 'hmclearn'
mcmc_neff_data(object, burnin = NULL, lagmax = NULL, ...)

mcmc_violin(object, ...)

## S3 method for class 'hmclearn'
mcmc_violin(object, burnin = NULL, ...)
```

Arguments

object	an object of class hmclearn, usually a result of a call to mh or hmc
...	optional additional arguments to pass to the bayesplot functions
burnin	optional numeric parameter for the number of initial MCMC samples to omit from the summary
lagmax	maximum lag to extract for determining effective sample sizes

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, burnin=NULL, ...)‘ Default plot called by ‘plot‘ function. Histograms of posterior draws with all chains merged.

‘mcmc_dens(object, burnin=NULL, ...)‘ Kernel density plots of posterior draws with all chains merged.

‘mcmc_hist_by_chain(object, burnin=NULL, ...)‘ Histograms of posterior draws with chains separated via faceting.

‘mcmc_dens_overlay(object, burnin=NULL, ...)‘ Kernel density plots of posterior draws with chains separated but overlaid on a single plot.

‘mcmc_violin(object, burnin=NULL, ...)‘ The density estimate of each chain is plotted as a violin with horizontal lines at notable quantiles.

‘mcmc_dens_chains(object, burnin=NULL, ...)‘ 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, burnin=NULL, ...)‘ Plots of uncertainty intervals computed from posterior draws with all chains merged.

‘mcmc_areas(object, burnin=NULL, ...)‘ Density plots computed from posterior draws with all chains merged, with uncertainty intervals shown as shaded areas under the curves.

‘mcmc_scatter(object, burnin=NULL, ...)‘ 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, burnin=NULL, ...)‘ 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, burnin=NULL, ...)‘ 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, burnin=NULL, ...)‘, ‘mcmc_rhat_hist(object, burnin=NULL, ...)‘ 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, burnin=NULL, ...)‘, ‘mcmc_neff_hist(object, burnin=NULL, ...)‘ 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, burnin=NULL, ...)‘, ‘mcmc_acf_bar(object, burnin=NULL, ...)‘ 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

```
# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)
```

```
f <- hmc(N = 1000,
          theta.init = rep(0, 3),
          epsilon = c(0.03, 0.02, 0.015),
          L = 10,
          logPOSTERIOR = poisson_posterior,
          glogPOSTERIOR = g_poisson_posterior,
          varnames = paste0("beta", 0:2),
          param = list(y=y, X=X),
          parallel=FALSE, chains=2)

mcmc_trace(f, burnin=100)
mcmc_hist(f, burnin=100)
mcmc_intervals(f, burnin=100)
mcmc_rhat(f, burnin=100)
mcmc_violin(f, burnin=100)
```

`leapfrog` *Leapfrog Algorithm for Hamiltonian Monte Carlo*

Description

Runs a single iteration of the leapfrog algorithm. Typically called directly from `hmc`

Usage

```
leapfrog(
  theta_lf,
  r,
  epsilon,
  glogPOSTERIOR,
  Minv,
  constrain,
  lastSTEP = FALSE,
  ...
)
```

Arguments

`theta_lf`	starting parameter vector
`r`	starting momentum vector
`epsilon`	Step-size parameter for `leapfrog`
`glogPOSTERIOR`	Function to calculate and return the gradient of the log posterior given a vector of values of `theta`
`Minv`	Inverse Mass matrix
`constrain`	Optional vector of which parameters in `theta` accept positive values only. Default is that all parameters accept all real numbers

lastSTEP	Boolean indicating whether to calculate the last half-step of the momentum update
...	Additional parameters passed to glogPOSTERIOR

Value

List containing two elements: theta.new the ending value of theta and r.new the ending value of the momentum

References

Neal, Radford. 2011. *MCMC Using Hamiltonian Dynamics.* In Handbook of Markov Chain Monte Carlo, edited by Steve Brooks, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng, 116–62. Chapman; Hall/CRC.

Examples

```
set.seed(321)
X <- cbind(1, rnorm(10))
y <- rnorm(10)
p <- runif(3) - 0.5
leapfrog(rep(0,3), p, 0.01, g_linear_posterior,
         diag(3), FALSE, X=X, y=y)
```

---

mh *Fit a generic model using Metropolis-Hastings (MH)*

Description

This function runs the MH algorithm on a generic model provided the logPOSTERIOR function. All parameters specified within the list param are passed to these the posterior function.

Usage

```
mh(
  N,
  theta.init,
  qPROP,
  qFUN,
  logPOSTERIOR,
  nu = 0.001,
  varnames = NULL,
  param = list(),
  chains = 1,
  parallel = FALSE,
  ...
)
```

Arguments

N	Number of MCMC samples
theta.init	Vector of initial values for the parameters
qPROP	Function to generate proposal
qFUN	Probability for proposal function. First argument is where to evaluate, and second argument is the conditional parameter
logPOSTERIOR	Function to calculate and return the log posterior given a vector of values of theta
nu	Single value or vector parameter passed to qPROP or qFUN for the proposal density
varnames	Optional vector of theta parameter names
param	List of additional parameters for logPOSTERIOR and glogPOSTERIOR
chains	Number of MCMC chains to run
parallel	Logical to set whether multiple MCMC chains should be run in parallel
...	Additional parameters for logPOSTERIOR

Value

Object of class hmclearn

Elements for hmclearn objects

N Number of MCMC samples

theta Nested list of length N of the sampled values of theta for each chain

thetaCombined List of dataframes containing sampled values, one for each chain

r NULL for Metropolis-Hastings

theta.all Nested list of all parameter values of theta sampled prior to accept/reject step for each

r.all NULL for Metropolis-Hastings

accept Number of accepted proposals. The ratio accept / N is the acceptance rate

accept_v Vector of length N indicating which samples were accepted

M NULL for Metropolis-Hastings

algorithm MH for Metropolis-Hastings

varnames Optional vector of parameter names

chains Number of MCMC chains

Available logPOSTERIOR functions

linear_posterior Linear regression: log posterior

logistic_posterior Logistic regression: log posterior

poisson_posterior Poisson (count) regression: log posterior

lmm_posterior Linear mixed effects model: log posterior

glmm_bin_posterior Logistic mixed effects model: log posterior

glmm_poisson_posterior Poisson mixed effects model: log posterior

Author(s)

Samuel Thomas <samthoma@iu.edu>, Wanzhu Tu <wtu@iu.edu>

Examples

```
# Linear regression example
set.seed(521)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

f1_mh <- mh(N = 3e3,
         theta.init = c(rep(0, 4), 1),
         nu <- c(rep(0.001, 4), 0.1),
         qPROP = qprop,
         qFUN = qfun,
         logPOSTERIOR = linear_posterior,
         varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
         param=list(y=y, X=X), parallel=FALSE, chains=1)

summary(f1_mh, burnin=1000)
```

---

mh.fit *Fitter function for Metropolis-Hastings (MH)*

Description

This is the basic computing function for MH and should not be called directly except by experienced users.

Usage

```
mh.fit(
  N,
  theta.init,
  qPROP,
  qFUN,
  logPOSTERIOR,
  nu = 0.001,
  varnames = NULL,
  param = list(),
  ...
)
```

Arguments

N	Number of MCMC samples
theta.init	Vector of initial values for the parameters
qPROP	Function to generate proposal
qFUN	Probability for proposal function. First argument is where to evaluate, and second argument is the conditional parameter
logPOSTERIOR	Function to calculate and return the log posterior given a vector of values of theta
nu	Single value or vector parameter passed to qPROP or qFUN for the proposal density
varnames	Optional vector of theta parameter names
param	List of additional parameters for logPOSTERIOR
...	Additional parameters for logPOSTERIOR

Value

List for mh

Elements in mh list

N Number of MCMC samples

theta Nested list of length N of the sampled values of theta for each chain

thetaCombined List of dataframes containing sampled values, one for each chain

r NULL for Metropolis-Hastings

theta.all Nested list of all parameter values of theta sampled prior to accept/reject step for each

r.all NULL for Metropolis-Hastings

accept Number of accepted proposals. The ratio accept / N is the acceptance rate

accept_v Vector of length N indicating which samples were accepted

M NULL for Metropolis-Hastings

algorithm MH for Metropolis-Hastings

Examples

```
# Logistic regression example
X <- cbind(1, seq(-100, 100, by=0.25))
betavals <- c(-0.9, 0.2)
lodds <- X %*% betavals
prob1 <- as.numeric(1 / (1 + exp(-lodds)))

set.seed(9874)
y <- sapply(prob1, function(xx) {
  sample(c(0, 1), 1, prob=c(1-xx, xx))
})
```

```
f1 <- mh.fit(N = 2000,
         theta.init = rep(0, 2),
         nu = c(0.03, 0.001),
         qPROP = qprop,
         qFUN = qfun,
         logPOSTERIOR = logistic_posterior,
         varnames = paste0("beta", 0:1),
         y=y, X=X)

f1$accept / f1$N
```

`neff` *Effective sample size calculation*

Description

Calculates an estimate of the adjusted MCMC sample size per parameter adjusted for autocorrelation.

Usage

```
neff(object, burnin = NULL, lagmax = NULL, ...)
```

Arguments

`object`	an object of class `hmclearn`, usually a result of a call to `mh` or `hmc`
`burnin`	optional numeric parameter for the number of initial MCMC samples to omit from the summary
`lagmax`	maximum lag to extract for determining effective sample sizes
`...`	currently unused

Value

Numeric vector with effective sample sizes for each parameter in the model

References

Gelman, A., et. al. (2013) *Bayesian Data Analysis*. Chapman and Hall/CRC. Section 11.5

Examples

```
# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)
```

```
f <- hmc(N = 1000,
          theta.init = rep(0, 3),
          epsilon = c(0.03, 0.02, 0.015),
          L = 10,
          logPOSTERIOR = poisson_posterior,
          glogPOSTERIOR = g_poisson_posterior,
          varnames = paste0("beta", 0:2),
          param = list(y=y, X=X),
          parallel=FALSE, chains=2)

neff(f, burnin=100)
```

`neff.hmclearn` *Effective sample size calculation*

Description

Calculates an estimate of the adjusted MCMC sample size per parameter adjusted for autocorrelation.

Usage

```
## S3 method for class 'hmclearn'
neff(object, burnin = NULL, lagmax = NULL, ...)
```

Arguments

`object`	an object of class `hmclearn`, usually a result of a call to `mh` or `hmc`
`burnin`	optional numeric parameter for the number of initial MCMC samples to omit from the summary
`lagmax`	maximum lag to extract for determining effective sample sizes
`...`	currently unused

Value

Numeric vector with effective sample sizes for each parameter in the model

References

Gelman, A., et. al. (2013) *Bayesian Data Analysis*. Chapman and Hall/CRC. Section 11.5

Examples

```
# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)

f <- hmc(N = 1000,
          theta.init = rep(0, 3),
          epsilon = c(0.03, 0.02, 0.015),
          L = 10,
          logPOSTERIOR = poisson_posterior,
          glogPOSTERIOR = g_poisson_posterior,
          varnames = paste0("beta", 0:2),
          param = list(y=y, X=X),
          parallel=FALSE, chains=2)

neff(f, burnin=100)
```

`plot.hmclearn` *Plot Histograms of the Posterior Distribution*

Description

Calls `mcmc_hist` from the `bayesplot` package to display histograms of the posterior

Usage

```
## S3 method for class 'hmclearn'
plot(x, burnin = NULL, ...)
```

Arguments

`x`	an object of class `hmclearn`, usually a result of a call to `mh` or `hmc`
`burnin`	optional numeric parameter for the number of initial MCMC samples to omit from the summary
`...`	optional additional arguments to pass to the `bayesplot` functions

Value

Calls `mcmc_hist` from the `bayesplot` package, which returns a list including a `ggplot2` object.

References

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

Examples

```
# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)

f <- hmc(N = 1000,
          theta.init = rep(0, 3),
          epsilon = c(0.03, 0.02, 0.015),
          L = 10,
          logPOSTERIOR = poisson_posterior,
          glogPOSTERIOR = g_poisson_posterior,
          varnames = paste0("beta", 0:2),
          param = list(y=y, X=X),
          parallel=FALSE, chains=2)

plot(f, burnin=100)
```

---

predict.hmclearn *Model Predictions for HMC or MH*

Description

`predict` generates simulated data from the posterior predictive distribution. This simulated data can be used for posterior predictive check diagnostics from the `bayesplot` package

Usage

```
## S3 method for class 'hmclearn'
predict(object, X, fam = "linear", burnin = NULL, draws = NULL, ...)
```

Arguments

`object`	an object of class `hmclearn`, usually a result of a call to `mh` or `hmc`
`X`	design matrix, either from fitting the model or new data
`fam`	generalized linear model family. Currently "linear", "binomial", and "poisson" are supported
`burnin`	optional numeric parameter for the number of initial MCMC samples to omit from the summary
`draws`	Number of simulated values from the posterior conditioned on `X`
`...`	additional parameters, currently unsupported

Value

An object of class `hmclearnpred`.

Elements of hmclearnpred objects

y Median simulated values for each observation in X

yrep Matrix of simulated values where each row is a draw from the posterior predictive distribution

X Numeric design matrix

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.

Examples

```
# Linear regression example
set.seed(521)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

f1 <- hmc(N = 500,
          theta.init = c(rep(0, 4), 1),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = linear_posterior,
          glogPOSTERIOR = g_linear_posterior,
          varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
          param=list(y=y, X=X), parallel=FALSE, chains=1)

summary(f1)

p <- predict(f1, X)
predvals <- p$y
plot(predvals, y, xlab="predicted", ylab="actual")

X2 <- cbind(1, matrix(rnorm(30), ncol=3))
p2 <- predict(f1, X2)
p2$y
```

---

psrf *Calculates Potential Scale Reduction Factor (psrf), also called the Rhat statistic, from models fit via* mh *or* hmc

---

Description

Gelman and Rubin's diagnostic assesses the mix of multiple MCMC chain with different initial parameter values Values close to 1 indicate that the posterior simulation has sufficiently converged, while values above 1 indicate that additional samples may be necessary to ensure convergence. A general guideline suggests that values less than 1.05 are good, between 1.05 and 1.10 are ok, and above 1.10 have not converged well.

Usage

```
psrf(object, burnin, ...)
```

Arguments

`object`	an object of class `hmclearn`, usually a result of a call to `mh` or `hmc`
`burnin`	optional numeric parameter for the number of initial MCMC samples to omit from the summary
`...`	currently unused

Value

Numeric vector of Rhat statistics for each parameter

References

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

Examples

```
# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)

f <- hmc(N = 1000,
          theta.init = rep(0, 3),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = poisson_posterior,
          glogPOSTERIOR = g_poisson_posterior,
          varnames = paste0("beta", 0:2),
          param = list(y=y, X=X),
          parallel=FALSE, chains=2)

psrf(f, burnin=100)
```

psrf.hmclearn *Calculates Potential Scale Reduction Factor (psrf), also called the Rhat statistic, from models fit via* mh *or* hmc

Description

Gelman and Rubin's diagnostic assesses the mix of multiple MCMC chain with different initial parameter values Values close to 1 indicate that the posterior simulation has sufficiently converged, while values above 1 indicate that additional samples may be necessary to ensure convergence. A general guideline suggests that values less than 1.05 are good, between 1.05 and 1.10 are ok, and above 1.10 have not converged well.

Usage

```
## S3 method for class 'hmclearn'
psrf(object, burnin = NULL, ...)
```

Arguments

object	an object of class hmclearn, usually a result of a call to mh or hmc
burnin	optional numeric parameter for the number of initial MCMC samples to omit from the summary
...	currently unused

Value

Numeric vector of Rhat statistics for each parameter

References

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

Examples

```
# poisson regression example
set.seed(7363)
X <- cbind(1, matrix(rnorm(40), ncol=2))
betavals <- c(0.8, -0.5, 1.1)
lmu <- X %*% betavals
y <- sapply(exp(lmu), FUN = rpois, n=1)

f <- hmc(N = 1000,
         theta.init = rep(0, 3),
```

```
            epsilon = 0.01,
            L = 10,
            logPOSTERIOR = poisson_posterior,
            glogPOSTERIOR = g_poisson_posterior,
            varnames = paste0("beta", 0:2),
            param = list(y=y, X=X),
            parallel=FALSE, chains=2)

psrf(f, burnin=100)
```

---

qfun *Multivariate Normal Density of Theta1 | Theta2*

Description

Provided for Random Walk Metropolis algorithm

Usage

```
qfun(theta1, theta2, nu)
```

Arguments

theta1	Vector of current quantiles
theta2	Vector for mean parameter
nu	Either a single numeric value for the covariance matrix, or a vector for the diagonal

Value

Multivariate normal density vector log-transformed

References

Alan Genz, Frank Bretz, Tetsuhisa Miwa, Xuefei Mi, Friedrich Leisch, Fabian Scheipl and Torsten Hothorn (2019). *mvtnorm: Multivariate Normal and t Distributions*

Examples

```
qfun(0, 0, 1)
log(1/sqrt(2*pi))
```

qprop *Simulate from Multivariate Normal Density for Metropolis Algorithm*

Description

Provided for Random Walk Metropolis algorithm

Usage

```
qprop(theta1, nu)
```

Arguments

theta1	Vector of current quantiles
nu	Either a single numeric value for the covariance matrix, or a vector for the diagonal

Value

Returns a single numeric simulated value from a Normal distribution or vector of length `theta1`. `length(mu)` matrix with one sample in each row.

References

B. D. Ripley (1987) *Stochastic Simulation*. Wiley. Page 98

Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S*. Fourth edition. Springer.

Examples

```
s <- replicate(1000, qprop(0, 1))
summary(s)
hist(s, col='light blue')
```

summary.hmclearn *Summarizing HMC Model Fits*

Description

summary method for class `hmclearn`

Usage

```
## S3 method for class 'hmclearn'
summary(
  object,
  burnin = NULL,
  probs = c(0.025, 0.05, 0.25, 0.5, 0.75, 0.95, 0.975),
  ...
)
```

Arguments

object	an object of class hmclearn, usually a result of a call to mh or hmc
burnin	optional numeric parameter for the number of initial MCMC samples to omit from the summary
probs	quantiles to summarize the posterior distribution
...	additional arguments to pass to quantile

Value

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

References

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

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

Examples

```
# Linear regression example
set.seed(521)
X <- cbind(1, matrix(rnorm(300), ncol=3))
betavals <- c(0.5, -1, 2, -3)
y <- X%*%betavals + rnorm(100, sd=.2)

f1 <- hmc(N = 500,
          theta.init = c(rep(0, 4), 1),
          epsilon = 0.01,
          L = 10,
          logPOSTERIOR = linear_posterior,
          glogPOSTERIOR = g_linear_posterior,
          varnames = c(paste0("beta", 0:3), "log_sigma_sq"),
          param=list(y=y, X=X), parallel=FALSE, chains=1)

summary(f1)
```

Index