Package 'XDNUTS'

September 15, 2025

Type Package

Title Discontinuous Hamiltonian Monte Carlo with Varying Trajectory Length

Version 1.6.5

Date 2025-09-13

Description Hamiltonian Monte Carlo for both continuous and discontinuous posterior distributions with a customizable trajectory length termination criterion. See Nishimura et al. (2020) <doi:10.1093/biomet/asz083> for the original Discontinuous Hamiltonian Monte Carlo; Hoffman et al. (2014) <doi:10.48550/arXiv.1111.4246> and Betancourt (2016) <doi:10.48550/arXiv.1601.00225> for the definition of possible Hamiltonian Monte Carlo termination criteria.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 2.10)

Imports base, coda, ggplot2, graphics, grDevices, gridExtra, parallel, purrr, Rcpp (>= 1.0.12), Rdpack, stats, MASS

LinkingTo Rcpp, RcppArmadillo

RdMacros Rdpack

Suggests knitr, rmarkdown, KernSmooth,

VignetteBuilder knitr

RoxygenNote 7.3.1

NeedsCompilation yes

Author Paolo Manildo [aut, cre] (ORCID: <https://orcid.org/0009-0006-7056-7012>)

Maintainer Paolo Manildo <paolo.manildo@phd.unipd.it>

Repository CRAN

Date/Publication 2025-09-14 23:00:08 UTC

Contents

`XDNUTS-package` *Discontinuous Hamiltonian Monte Carlo with Varying Trajectory Length*

Description

Hamiltonian Monte Carlo for both continuous and discontinuous posterior distributions with a customizable trajectory length termination criterion. See Nishimura et al. (2020) <doi:10.1093/biomet/asz083> for the original Discontinuous Hamiltonian Monte Carlo; Hoffman et al. (2014) <doi:10.48550/arXiv.1111.4246> and Betancourt (2016) <doi:10.48550/arXiv.1601.00225> for the definition of possible Hamiltonian Monte Carlo termination criteria.

Details

The DESCRIPTION file:

Package:	XDNUTS
Type:	Package
Title:	Discontinuous Hamiltonian Monte Carlo with Varying Trajectory Length
Version:	1.6.5
Date:	2025-09-13
Authors@R:	person(given = "Paolo", family = "Manildo", email = "paolo.manildo@phd.unipd.it", role = c("aut", "cre")
Description:	Hamiltonian Monte Carlo for both continuous and discontinuous posterior distributions with a customizabl
License:	MIT + file LICENSE
Encoding:	UTF-8
LazyData:	true
Depends:	R (>= 2.10)
Imports:	base, coda, ggplot2, graphics, grDevices, gridExtra, parallel, purrr, Rcpp (>= 1.0.12), Rdpack, stats, MASS
LinkingTo:	Rcpp, RcppArmadillo
RdMacros:	Rdpack
Suggests:	knitr, rmarkdown, KernSmooth,

VignetteBuilder:	knitr
RoxygenNote:	7.3.1
Author:	Paolo Manildo [aut, cre] (<https://orcid.org/0009-0006-7056-7012>)
Maintainer:	Paolo Manildo <paolo.manildo@phd.unipd.it>

Index of help topics:

The package allows to use a more efficient version of the Discontinuous Hamiltonian Monte Carlo proposed in (Nishimura et al. 2020), thanks to the use of recycled samples from each trajectory (Nishimura and Dunson 2020) and a termination criterion for identyfing the optimal discrete integration time of each trajectory (Betancourt 2016). No models are at disposal, so the user must specify one through the definition of the function `nlp`. This function must evaluate the negative log posterior of the model and its gradient with respect to the first $d - k$ parameters. d is the model dimension, while k is the number of parameters for which the sampling scheme will be based on the method described in (Nishimura et al. 2020). This method was born for treating discontinuous components but it is applicable to continuous one too. `nlp` must be a function with 3 arguments:

- the vector of parameters, current state of the chain/trajectory, for which the negative log posterior or its gradient must be evaluated.
- a list object that contains the necessary argouments, namely data and hyperparameters.

- a boolean value, TRUE to evaluate only the negative log posterior of the models, FALSE to evaluate its gradient with respect to the continuous components of the posterior.

The available algorithms are the following

NUTS No U-Turn Sampler of (Hoffman et al. 2014).

XHMC Hamiltonian Monte Carlo with a termination criterion based on the exhustion of the virial (Betancourt 2016) which require the specification of a threshold.

HMC Hamiltonian Monte Carlo with trajectory length varying uniformly inside a user specified interval. Instead of proposing the last value of each trajectories a sample is drawn uniformly from them. Reference can be found in (Betancourt 2017).

All of them are embedded into the framework described in (Nishimura et al. 2020) which allows the use of Hamiltonian Monte Carlo with discontinuous posterior and hence to discrete parameter space by the definition of a step function shape density.

Author(s)

Paolo Manildo [aut, cre] (<https://orcid.org/0009-0006-7056-7012>)

Maintainer: Paolo Manildo <paolo.manildo@phd.unipd.it>

References

Hoffman MD, Gelman A, others (2014). "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." *J. Mach. Learn. Res.*, **15**(1), 1593–1623.

Betancourt M (2016). "Identifying the optimal integration time in Hamiltonian Monte Carlo." *arXiv preprint arXiv:1601.00225*.

Betancourt M (2017). "A conceptual introduction to Hamiltonian Monte Carlo." *arXiv preprint arXiv:1701.02434*.

Nishimura A, Dunson DB, Lu J (2020). "Discontinuous Hamiltonian Monte Carlo for discrete parameters and discontinuous likelihoods." *Biometrika*, **107**(2), 365–380.

Nishimura A, Dunson D (2020). "Recycling Intermediate Steps to Improve Hamiltonian Monte Carlo." *Bayesian Analysis*, **15**(4). ISSN 1936-0975, doi:10.1214/19ba1171, http://dx.doi.org/10.1214/19-BA1171.

`main_function` *Function to generate a Markov chain for both continuous and discontinuous posterior distributions.*

Description

The function allows to generate a single Markov Chain for sampling from both continuous and discontinuous posterior distributions using a plethora of algorithms. Classic Hamiltonian Monte Carlo (Duane et al. 1987) , NUTS (Hoffman et al. 2014) and XHMC (Betancourt 2016) are embedded into the framework described in (Nishimura et al. 2020), which allows to deal with such posteriors. Furthermore, for each method, it is possible to recycle samples from the trajectories using the method proposed by (Nishimura and Dunson 2020). This is used to improve the estimation of the mass matrix during the warm-up phase without requiring significant additional computational costs. This function should not be used directly, but only through the user interface provided by xdnuts.

Usage

```
main_function(
  theta0,
  nlp,
  args,
  k,
  N,
  K,
  tau,
  L,
  thin,
  chain_id,
  verbose,
  control
)
```

Arguments

`theta0`	a vector of length-d containing the starting point of the chain.
`nlp`	a function object that takes three arguments: **par** a vector of length-d containing the value of the parameters. **args** a list object that contains the necessary arguments, namely data and hyperparameters. **eval_nlp** a boolean value, `TRUE` for evaluating only the model\'s negative log posterior, `FALSE` to evaluate the gradient with respect to the continuous components of the posterior.
`args`	the necessary arguments to evaluate the negative log posterior and its gradient.
`k`	the number of parameters that induce a discontinuity in the posterior distribution.
`N`	integer containing the number of post warm-up samples to evaluate.
`K`	integer containing the number of recycled samples from each trajectory during the warm-up phase or beyond.
`tau`	the threshold for the exhaustion termination criterion described in (Betancourt 2016).

`L`	the desired length of the trajectory of classic Hamiltonian Monte Carlo algorithm.
`thin`	integer containing the number of samples to discard in order to produce a final iteration of the chain.
`chain_id`	the identification number of the chain.
`verbose`	a boolean value that controls whether to print all the information regarding the sampling process.
`control`	an object of class `control_xdnuts` containing the specifications for the algorithm. See the set_parameters function for detail.

Value

a named list containing:

values a $N \times d$ matrix containing the sample from the target distribution (if convergence has been reached).

energy a vector of length-N containing the Markov Chain of the energy level sets.

delta_energy a vector of length-N containing the Markov Chain of the first difference energy level sets.

step_size a vector of length-N containing the sampled step size used for each iteration.

step_length a vector of length-N containing the length of each trajectory of the chain.

alpha a vector of length-$k + 1$ containing the estimate of the Metropolis acceptance probabilities. The first element of the vector is the estimated global acceptance probability. The remaining k elements are the estimate rate of reflection for each parameters which travels coordinate-wise through some discontinuity.

warm_up a $N_{adapt} \times d$ matrix containing the sample of the chain coming from the warm-up phase. If `keep_warm_up = FALSE` inside the `control` argument, nothing is returned.

div_trans a $M \times d$ matrix containing the locations where a divergence has been encountered during the integration of Hamilton equation. Hopefully $M \ll N$, and even better if $M = 0$.

M_cont the Mass Matrix of the continuous components estimated during the warm-up phase. Based on the `M_type` value of the `control` arguments, this could be an empty object, a vector or a matrix.

M_disc the Mass Matrix of the discontinuous components estimated during the warm-up phase. Based on the `M_type` value of the `control` arguments, this could be an empty object or a vector.

References

Betancourt M (2016). "Identifying the optimal integration time in Hamiltonian Monte Carlo." *arXiv preprint arXiv:1601.00225*.

Duane S, Kennedy AD, Pendleton BJ, Roweth D (1987). "Hybrid monte carlo." *Physics letters B*, **195**(2), 216–222.

Hoffman MD, Gelman A, others (2014). "The No-U-Turn sampler: adaptively setting path lengths

in Hamiltonian Monte Carlo." *J. Mach. Learn. Res.*, **15**(1), 1593–1623.

Nishimura A, Dunson D (2020). "Recycling Intermediate Steps to Improve Hamiltonian Monte Carlo." *Bayesian Analysis*, **15**(4). ISSN 1936-0975, doi:10.1214/19ba1171, http://dx.doi.org/10.1214/19-BA1171.

Nishimura A, Dunson DB, Lu J (2020). "Discontinuous Hamiltonian Monte Carlo for discrete parameters and discontinuous likelihoods." *Biometrika*, **107**(2), 365–380.

`plot.XDNUTS` *Function to view the draws from the posterior distribution.*

Description

Function to view the draws from the posterior distribution.

Usage

```
## S3 method for class 'XDNUTS'
plot(
  x,
  type = 1,
  which = NULL,
  warm_up = FALSE,
  plot.new = FALSE,
  which_chains = NULL,
  colors = NULL,
  gg = TRUE,
  scale = 1,
  ...
)
```

Arguments

`x`	an object of class `XDNUTS`.
`type`	the type of plot to display.
	`type = 1` marginal chains, one for each desired dimension.
	`type = 2` bivariate plot.
	`type = 3` time series plot of the energy level sets. Good for a quick diagnostic of big models.
	`type = 4` stickplot of the step-length of each iteration.
	`type = 5` Histograms of the centered marginal energy in gray and of the first differences of energy in red.
	`type = 6` Autoregressive function plot of the parameters.
	`type = 7` Matplot of the empirical acceptance rate and refraction rates.

which	either a numerical vector indicating the index of the parameters of interest or a string which = 'continuous' for plotting the first $d - k$ parameters. which = 'discontinuous' for plotting the last k parameters. where both d and k are elements contained in the output of the xdnuts function. If type = 7, it refers to the rates index instead. When which = 'continuous', only the global acceptance rate is displayed. In contrast, when which = 'discontinuous', the refraction rates are shown.
warm_up	a boolean value indicating whether the plot should be made using the warm-up samples.
plot.new	a boolean value indicating whether a new graphical window should be opened. This is advised if the parameters space is big.
which_chains	a numerical vector indicating the index of the chains of interest.
colors	a numerical vector containing the colors for each chain specified in which_chains or for each rate specified in which when type = 7.
gg	A boolean value indicating whether the plot should utilize the grammar of graphics features. Default value is set to TRUE.
scale	A numeric value for scaling the appearance of plots.
...	additional arguments to customize plots. In reality, these do nothing.

Value

A graphical object if gg = TRUE, otherwise nothing is returned.

`print.summary.XDNUTS` *Function for printing an object of class summary.XDNUTS*

Description

Print to console the statistics about the MCMC samples obtained with a call to the function summary.XDNUTS. See that for details.

Usage

```
## S3 method for class 'summary.XDNUTS'
print(x, ..., digits = 3)
```

Arguments

x	an object of class summary.XDNUTS
...	additional values to pass. These currently do nothing.
digits	number of digits for rounding the output. Default value is 3.

Value

No return value.

print.XDNUTS *Function for printing an object of class XDNUTS*

Description

Print to console the specific of the algorithm used to generate an XDNUTS object.

Usage

```
## S3 method for class 'XDNUTS'
print(x, ..., digits = 3, show_all = FALSE)
```

Arguments

x	an object of class XDNUTS.
...	additional arguments to pass. These currently do nothing.
digits	a numeric scalar indicating the number of digits to show. Default value is 3.
show_all	logical scalar indicating where to print all the summary statistics even if these are more than 10.

Value

Return a data frame object.

set_parameters *Function that regulates the specifications of the xdnuts function.*

Description

Function that regulates the specifications of the xdnuts function.

Usage

```
set_parameters(
  N_init1 = 50L,
  N_adapt = 200L,
  N_init2 = 75L,
  burn_adapt_ratio = 0.1,
  keep_warm_up = FALSE,
  recycle_only_init = TRUE,
  max_treedepth = 10L,
  max_treedepth_init = 10L,
  eps_jitter = 0.1,
  L_jitter = 3L,
  gamma = 0.05,
```

```
  kappa = 0.75,
  delta = NULL,
  t0 = 10L,
  M_type = "dense",
  refresh = 0.1,
  l_eps_init = NA_real_,
  different_stepsize = FALSE,
  mu = NA_real_,
  M_cont = NULL,
  M_disc = NULL
)
```

Arguments

`N_init1`	an integer that regulates the number of samples used to adapt the step size.
`N_adapt`	an integer that regulates the number of samples used to estimate the Mass Matrix with fixed step size.
`N_init2`	an integer that regulates the number of samples used to adapt the step size after the estimation of the Mass Matrix.
`burn_adapt_ratio`	a numeric scalar $\in (0, 1]$ indicating the ratio of warm-up samples to discard in order to estimate the covariance matrix of the parameters.
`keep_warm_up`	a logical scalar that determines whether the warm-up samples should be returned.
`recycle_only_init`	a logical value which disables the recycling of the samples from each trajectory once the warm-up phase has terminated.
`max_treedepth`	an integer that regulates the maximum depth of the binary tree used to approximate Hamilton equation for the exploration of each energy level set of the phase space.
`max_treedepth_init`	an integer that controls the maximum depth of the binary tree during the step-size adaptation phase. Setting a smaller value can help avoid wasting valuable time on low-probability areas with suboptimal algorithm parameters.
`eps_jitter`	a numeric scalar which regulates the amount of jittering used to perturb the value of the step size for each iteration of the chain after the warm-up phase.
`L_jitter`	an integer scalar that regulates the amount of jittering used to perturb the value of the trajectory length if this is specified to be constant. This occurs when the classic Hamiltonian Monte Carlo algorithm is used through the `method = "HMC"` option in the xdnuts function. If `L_jitter` ≥ 1 each trajectory length is sampled uniformly inside the interval [L - L_jitter , L + L_jitter].
`gamma`	a numeric value that, in the Nesterov Dual Averaging algorithm, regulates the sensitivity of the step size updating scheme to fluctuations in the estimate of the mean Metropolis acceptance probability.
`kappa`	a numeric value that regulates the vanishing of Nesterov Dual Averaging algorithm for the estimation of the step size.

`delta` a vector containing the Metropolis acceptance probabilities, including both the global and those related to potential differences. Default values are (0.8,0.6). If the second element of the vector is set to `NA`, then the step size calibration is conducted solely through the global acceptance probabilities.

`t0` an integer value that makes Nesterov Dual Averaging algorithm for the estimation of the step size less sensitive to early iterations.

`M_type` a character value specifying the type of Mass Matrix to estimate:

- `"identity"` no Mass Matrix estimation is done.
- `"diagonal"` a diagonal Mass Matrix is estimated during the warm-up phase.
- `"dense"` a full dense Mass Matrix is estimated during the warm-up phase.

`refresh` a numeric scalar bounded in $(0, 1)$ which regulates the update frequency of the displayed sampling process state. Default values is 0.1, meaning every 10% of the total samples.

`l_eps_init` a numeric scalar containing the logarithm of the initial value for the step size used to approximate Hamilton differential equation for phase space exploration.

`different_stepsize` a boolean value indicating where the adaptation scheme should adapt different step size. If `TRUE`, a global step size is adapted via Nesterov Dual Averaging algorithm. At the same time, for each empirical reflection rate of each component treated as discontinuous the same algorithm is exploited and the difference between these is obtained through the updating of the discontinuous components Mass Matrix. Default value is `FALSE`.

`mu` a numeric scalar containing the value to which the step size is shrunken during the warm-up phase.

`M_cont` a vector of length-$d - k$ if `M_type = "diagonal"` or a $(d - k) \times (d - k)$ matrix if `M_type = "dense"` containing an initial estimate for the Mass Matrix (the inverse of the parameters covariance matrix). If you want to keep it fixed, they should specify `N_adapt = 0`.

`M_disc` a vector of length-k if `M_type = "diagonal"` or `M_type = "dense"` containing an initial estimate for the Mass Matrix (the inverse of the parameters covariances). If one wants to keep it fixed, they should specify `N_adapt = 0`.

Value

an object of class `control_xdnuts` containing a named list with all the above parameters.

`summary.XDNUTS` *Function to print the summary of an XDNUTS model.*

Description

Function to print the summary of an XDNUTS model.

Usage

```
## S3 method for class 'XDNUTS'
summary(
  object,
  ...,
  q.val = c(0.05, 0.25, 0.5, 0.75, 0.95),
  which = NULL,
  which_chains = NULL
)
```

Arguments

`object`	an object of class `XDNUTS`.
`...`	additional arguments to customize the summary.
`q.val`	desired quantiles of the posterior distribution for each coordinate. Default values are `0.05,0.25,0.5,0.75,0.95`.
`which`	either a numerical vector indicating the index of the parameters of interest or a string `which = 'continuous'` for plotting the first $d - k$ parameters. `which = 'discontinuous'` for plotting the last k parameters. where both d and k are elements contained in the output of the function xdnuts.
`which_chains`	a numerical vector indicating the index of the chains of interest.

Value

a list containing a data frame named `stats` with the following columns:

mean the mean of the posterior distribution.

sd the standard deviation of the posterior distribution.

q.val the desired quantiles of the posterior distribution.

ESS the Effective Sample Size for each marginal distribution.

R_hat the Potential Scale Reduction Factor of Gelman (Gelman and Rubin 1992), only if multiple chains are available.

R_hat_upper_CI the upper confidence interval for the latter, only if multiple chains are available.

Other quantities returned are:

Gelman.Test the value of the multivariate Potential Scale Reduction Factor test (Gelman and Rubin 1992).

BFMI the value of the empirical Bayesian Fraction of Information Criteria (Betancourt 2016). A value below 0.2 indicates a bad random walk behavior in the energy Markov Chain, mostly due to a suboptimal specification of the momentum parameters probability density.

n_div the total number of trajectory ended with a divergent transition.

n_premature the total number of trajectory with a premature termination.

References

Betancourt M (2016). "Diagnosing suboptimal cotangent disintegrations in Hamiltonian Monte Carlo." *arXiv preprint arXiv:1604.00695*.

Gelman A, Rubin DB (1992). "Inference from iterative simulation using multiple sequences." *Statistical science*, **7**(4), 457–472.

`trajectories` *Function that approximate the Hamiltonian Flow for given starting values of the position and momentum of a particle in the phase space defined by the kinetic and potential energy provided in input.*

Description

Function that approximate the Hamiltonian Flow for given starting values of the position and momentum of a particle in the phase space defined by the kinetic and potential energy provided in input.

Usage

```
trajectories(theta0, m0, nlp, args, eps, k, M_cont, M_disc, max_it)
```

Arguments

`theta0`	a numeric vector of length d representing the starting position vector for the particle.
`m0`	a numeric vector of length d representing the starting momenta vector for the particle.
`nlp`	a function object that evaluate the negative of the logarithm of a probability density function, and its gradient, i.e. the potential energy function of the system.
`args`	a list object containing the arguments to be passed to the function `nlp`.
`eps`	a numeric scalar indicating the step size for the *leapfrog* integrator.
`k`	an integer scalar indicating the number of discontinuous components of `theta0`.
`M_cont`	either a vector or a squared matrix, of the same length/dimension of the position/momenta vector, representing the continuous components mass matrix.
`M_disc`	a vector of the same length of the position/momenta vector, representing the discontinuous components mass matrix.
`max_it`	an integer value indicating the length of the trajectory. This quantity times `eps` is equal to the approximated integration time of the Hamiltonian flow.

Value

a data frame that summarizes the approximated Hamiltonian flow.

- The first d columns contain the particle position evolution.
- The second d columns contain the particle momenta evolution.
- The $2d + 1$ column contains the Hamiltonian evolution.
- The $2d + 2$ column contains the evolution of the No U-Turn Sampler termination criterion.
- The $2d + 3$ column contains the evolution of the virial exhaustion termination criterion.
- Acceptance and or refraction probabilities. This depends on the value of `k`.
- Reflession dummy indicators.
- Divergent transition dummy indicators.

`viscosity` *Blood viscosity data*

Description

Contains blood viscosity measurements of 6 subjects. Each individual was measured 7 times.

Usage

```
viscosity
```

Format

A data frame with 6 rows and 8 variables:

id subject identifier.

Time.1 viscosity measurements for the first time.,

Time.2 viscosity measurements for the second time.,

Time.3 viscosity measurements for the third time.,

Time.4 viscosity measurements for the fourth time.,

Time.5 viscosity measurements for the fifth time.,

Time.6 viscosity measurements for the sixth time.,

Time.7 viscosity measurements for the seventh time.

Source

Master\'s Degree course at the University of Padua

Examples

```
data(viscosity)
```

`xdextract` *Function to extract samples from the output of an XDNUTS model.*

Description

Function to extract samples from the output of an XDNUTS model.

Usage

```
xdextract(
  X,
  which = NULL,
  which_chains = NULL,
  collapse = FALSE,
  thin = NULL,
  burn = NULL
)
```

Arguments

`X`	an object of class `XDNUTS`.
`which`	either a numerical vector indicating the index of the parameters of interest or a string `which = 'continuous'` for plotting the first $d - k$ parameters. `which = 'discontinuous'` for plotting the last k parameters. where both d and k are elements contained in the output of the function xdnuts.
`which_chains`	a vector of indices containing the chains to extract. By default, all chains are considered.
`collapse`	a boolean value. If TRUE, all samples from every chain are collapsed into one. The default value is FALSE.
`thin`	an integer value indicating how many samples should be discarded before returning an iteration of the chain.
`burn`	an integer value indicating how many initial samples for each chain to burn.

Value

an $N \times d$ matrix or an $N \times d \times C$ array, where C is the number of chains, containing the MCMC samples.

`xdnuts` *Discontinuous Hamiltonian Monte Carlo using both manual and automatic termination criteria.*

Description

The function allows generating multiple Markov Chains for sampling from both continuous and discontinuous posterior distributions using a variety of algorithms. Classic Hamiltonian Monte Carlo (Duane et al. 1987), NUTS (Hoffman et al. 2014), and XHMC (Betancourt 2016) are embedded into the framework described in (Nishimura et al. 2020), which allows dealing with such posteriors. Furthermore, for each method, it is possible to recycle samples from the trajectories using the method proposed by (Nishimura and Dunson 2020). This is used to improve the estimate of the Mass Matrix during the warm-up phase without requiring a relevant additional computational cost.

Usage

```
xdnuts(
  theta0,
  nlp,
  args,
  k,
  N = 1000,
  K = 3,
  method = "NUTS",
  tau = NULL,
  L = NULL,
  thin = 1,
  control = set_parameters(),
  parallel = FALSE,
  loadLibraries = NULL,
  loadRObject = NULL,
  verbose = FALSE,
  hide = FALSE,
  logfile = ""
)
```

Arguments

`theta0` a list containing the starting values for each chain. These starting values are vectors of length-d. The last $k \in [0, d]$ elements refer to parameters which determine a discontinuity in the posterior distribution.

`nlp` a function which evaluates the negative log posterior and its gradient with respect to parameters that do not induce any discontinuity in the posterior distribution (more generally, the first $d - k$ parameters). This function must take 3 arguments:

par a vector of length-d containing the parameter values.

args a list object that contains the necessary arguments, namely data and hyperparameters.

eval_nlp a boolean value, TRUE to evaluate only the negative log posterior of the models, FALSE to evaluate its gradient with respect to the continuous components of the posterior.

args — a list containing the inputs for the negative posterior function.

k — an integer value that states the number of parameters that determines a discontinuity in the posterior distribution. Actually, since the algorithm proposed in (Nishimura et al. 2020) also works for the full continuous case, k is the number of parameters specified by the user for which this algorithm is used.

N — the number of draws from the posterior distribution, after warm-up, for each chain. Default value is 1000.

K — the number of recycled samples per iteration used by default during the warm-up phase. Default value is 3. To recycle in the sampling phase too, specify recycle_only_init = FALSE in the control argument above.

method — a character value which defines the type of algorithm to exploit:

"NUTS" applies the No U-Turn Sampler of (Hoffman et al. 2014).

"XHMC" applies the Exhaustion Hamiltonian Monte Carlo of (Betancourt 2016).

"HMC" applies one of the classic version of Hamiltonian Monte Carlo algorithm, in particular the one described in (Betancourt 2017), which samples from the trajectory instead of always returning the last value.

tau — the threshold for the virial termination criterion (Betancourt 2016).

L — the desired length of the trajectory of classic Hamiltonian Monte Carlo algorithm.

thin — the number of necessary and discarded samples to obtain a final iteration of one chain.

control — an object of class control_xdnuts, output of the function set_parameters.

parallel — a boolean value specifying whether the chains must be run in parallel. Default value is FALSE.

loadLibraries — A character vector indicating the names of the packages to load on each cluster if parallel is set to TRUE.

loadRObject — A character vector indicating the names of the R objects to load on each cluster if parallel is set to TRUE.

verbose — a boolean value for printing all the information regarding the sampling process.

hide — a boolean value that omits the printing to the console if set to TRUE.

logfile — The pathname of the log file. The default value is "". On Linux or macOS systems, this allows the output to be printed directly to the console. Unfortunately, this is not possible on Windows systems.

Value

a list of class XDNUTS containing

chains a list of the same length of `theta0`, each element containing the output from the function main_function.

d the dimension of the parameter space.

k the number of parameters that lead to a discontinuous posterior distribution. Or, more generally, for which the algorithm of (Nishimura et al. 2020) is exploited.

K the number of recycled samples for each iteration during the sampling phase.

N the number of posterior draws for each chain.

method the MCMC method used. This could be either "NUTS", "XHMC", or "HMC".

tau the threshold for the virial termination criterion (Betancourt 2016). Only if `method = "XHMC"` this value is different from zero.

L the desired length of the trajectory of classic Hamiltonian Monte Carlo algorithm specified by the user. This argument is necessary if `method = "HMC"`.

thin the number of discarded samples for every final iteration, specified by the user.

control an object of class `control_xdnuts`, output of the function set_parameters with arguments specified by the user.

verbose the boolean value specified by the user regarding the printing of the sampling process information.

parallel the boolean value specified by the user regarding parallel processing.

References

Betancourt M (2016). "Identifying the optimal integration time in Hamiltonian Monte Carlo." *arXiv preprint arXiv:1601.00225*.

Betancourt M (2017). "A conceptual introduction to Hamiltonian Monte Carlo." *arXiv preprint arXiv:1701.02434*.

Duane S, Kennedy AD, Pendleton BJ, Roweth D (1987). "Hybrid monte carlo." *Physics letters B*, **195**(2), 216–222.

Hoffman MD, Gelman A, others (2014). "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." *J. Mach. Learn. Res.*, **15**(1), 1593–1623.

Nishimura A, Dunson D (2020). "Recycling Intermediate Steps to Improve Hamiltonian Monte Carlo." *Bayesian Analysis*, **15**(4). ISSN 1936-0975, doi:10.1214/19ba1171, http://dx.doi.org/10.1214/19-BA1171.

Nishimura A, Dunson DB, Lu J (2020). "Discontinuous Hamiltonian Monte Carlo for discrete parameters and discontinuous likelihoods." *Biometrika*, **107**(2), 365–380.

`xdtransform` *Function to apply a transformation to the samples from the output of an XDNUTS model.*

Description

Function to apply a transformation to the samples from the output of an XDNUTS model.

Usage

```
xdtransform(
  X,
  FUN = NULL,
  ...,
  which = NULL,
  which_chains = NULL,
  new.names = NULL,
  thin = NULL,
  burn = NULL
)
```

Arguments

`X`	an object of class `XDNUTS`.
`FUN`	a function object which takes one or more components of an MCMC iteration and any other possible arguments.
`...`	optional arguments for FUN.
`which`	a vector of indices indicating which parameter the transformation should be applied to. If `NULL`, the function is applied to all of them.
`which_chains`	a numerical vector indicating the index of the chains of interest.
`new.names`	a character vector containing the parameter names in the new parameterization. If only one value is provided, but the transformation involves more, the name is iterated with an increasing index.
`thin`	an integer value indicating how many samples should be discarded before returning an iteration of the chain.
`burn`	an integer value indicating how many initial samples for each chain to discard.

Value

an object of class `XDNUTS` with the specified transformation applied to each chain.

Index