Package: bayesian.simulations (via r-universe)

November 14, 2026

Type Package

Title Bayesian Methods for Calibrating Simulation Sets

Version 0.2.2

Author Who wrote it

Maintainer Anthony Todd Fojo <anthony.fojo@jhmi.edu>

Description More about what it does (maybe more than one line) Use four spaces when indenting paragraphs within the Description.

License What license is it under?

Encoding UTF-8

LazyData true

Collate 'parallelization.R' 'mcmc_class.R' 'transformations.R' 'simset_class.R' 'sampling.R' 'helpers.R' 'merge.R' 'cache.R' 'run_interface.R' 'adaptive_blockwise_metropolis.R' 'postprocessing.R' 'output.R'

Imports mvtnorm, ggplot2, parallel, reshape2, distributions, lhs

RoxygenNote 7.1.1

Config/pak/sysreqs libicu-dev

Repository https://cipher-epi.r-universe.dev

Date/Publication 2025-08-25 20:24:54 UTC

RemoteUrl https://github.com/tfojo1/bayesian.simulations

RemoteRef HEAD

RemoteSha 4e0d13e85857396bb0e6e2ac1d244775b2145f75

Contents

add.parameters 2
assemble.mcmc.from.cache 3
cache.progress 4
create.mcmc.cache 4
extend.simulations 5

extract.simset 5
extract.simset.distribution 6
extract.simset.parameter.distribution 6
get.rhats 7
is.mcmc.cache.complete 8
mcmc.merge.parallel 8
mcmc.merge.serial 8
pair.plot 9
process.transformations 9
remove.mcmc.cache 10
run.mcmc.from.cache 10
simset 11
subset.mcmc 11
subset.simset 12
thin.simset 13
trace.plot 13

Index **15**

`add.parameters` *Add parameters to a simset or parameter_set*

Description

Add parameters to a simset or parameter_set

Usage

```
add.parameters(
  simset,
  parameters,
  parameter.names = NULL,
  parameter.transformations = NULL,
  parameter.lower.bounds = -Inf,
  parameter.upper.bounds = Inf
)
```

Arguments

`simset`	A simset or parameter_set to which to add parameters
`parameters`	The parameters to add. If only one new parameter, can be either a vector of length simset@n.sim or length 1, or a matrix with one column and simset@n.sim rows. If more than one parameter is to be added, a matrix with one column for parameter and simset@n.sim rows
`parameter.names`	The names of the new parameters being added. If NULL, takes the parameter names from the column names of parameters

`parameter.transformations`
: Objects of class distributions::transformation, or the names of defined transformations, according to which parameters should be smoothed when making smoothed distributions

`parameter.lower.bounds`, `parameter.upper.bounds`
: The min and max values each parameter can take. If given a scalar value, will apply the same bounds to all parameters

---

`assemble.mcmc.from.cache` *Assemble an MCMC object from iterations stored in a cache*

Description

Assemble an MCMC object from iterations stored in a cache

Usage

```
assemble.mcmc.from.cache(
  dir,
  allow.incomplete = F,
  chains = NULL,
  remove.cache.when.done = F
)
```

Arguments

`dir`	The directory where the cache is stored
`allow.incomplete`	If FALSE, throws an error if the cache has not finished running. If TRUE, assembles an MCMC object out of all chains where any iterations have been run past the burn period, with a number of iterations equal to the mininum number of iterations run in any chain past the burn period
`chains`	The numbers of the chains to run. If passed NULL (the default), runs all chains

Value

An mcmc.sim object

cache.progress *Report the progress of an MCMC process running from a cache*

Description

Report the progress of an MCMC process running from a cache

Usage

```
cache.progress(dir)
```

Arguments

dir	The directory where the cache is stored

create.mcmc.cache *Set up a cache for running an MCMC from*

Description

Set up a cache for running an MCMC from

Usage

```
create.mcmc.cache(
  dir,
  control,
  n.iter,
  starting.values = NULL,
  prior.mcmc = NULL,
  cache.frequency = 1000,
  allow.overwrite.cache = F,
  merge.with.prior.mcmc = T
)
```

Arguments

dir	The directory where the cache should be stored
cache.frequency	How often to cache to the disk
allow.overwrite.cache	If a previous cache is in place in dir, the function will throw an error if allow.overwrite.cache is FALSE. If allow.overwrite.cache is TRUE, it will overwrite the previously existing cache
cache.prefix	???

extend.simulations *Extend the simulations of a simset*

Description

Extend the simulations of a simset

Usage

```
extend.simulations(simset, fn)
```

Arguments

simset	An object of class simset
fn	A function that takes two arguments: sim (the prior simulation object) and parameters (the corresponding numeric vector of parameters) and returns a new simulation object

Value

An updated simset

extract.simset *Extracts a simset from an object containing simulations*

Description

Extracts a simset from an object containing simulations

Usage

```
extract.simset(object, ...)
```

Arguments

object	The object from which to extract the simset

extract.simset.distribution

Extract a distribution of simulation results

Description

Gets a probability distribution over the results of a function applied to each simulation or its parameters or both

Usage

```
extract.simset.distribution(
  simset,
  fn,
  pass.to.fn = c("simulation", "parameters", "both")[1],
  ...,
  smooth = F
)
```

Arguments

simset	A simset object
fn	A function to be applied. If pass.to.fn=='simulation', the function's first parameters should be 'sim' (a simulation objet). If pass.to.fn=='parameters', the function's first argument should be 'parameters' (a vector of parameters). If pass.to.fn=='both', the function's first two parameters should be 'sim' and 'parameters'
pass.to.fn	What to pass to fn as arguments
...	Additional parameters to be passed to fn
smooth	Whether to return a smoothed distribution over continuous parameters

extract.simset.parameter.distribution

Extract the distribution of parameters in a simset

Description

Gets a probability distribution over the parameters in a simset

Usage

```
extract.simset.parameter.distribution(simset, smooth = F)
```

Arguments

simset	A simset object
smooth	Whether to return a smoothed distribution over continuous parameters

---

`get.rhats` *Calculate R-hat statistics for an MCMC object*

Description

Calculates the R-hat of Gelman 2013, or the R-hat using rank-normalization as per Vehtari 2019

Usage

```
get.rhats(
  mcmc,
  var.names = mcmc@var.names,
  chains = 1:mcmc@n.chains,
  additional.burn = 0,
  additional.thin = 1,
  exact.var.names = F,
  rank.normalize = F,
  sort = T
)
```

Arguments

mcmc	A mcmc.sim object
var.names	var.names is the names of variables to include or a set of integer indices. If exact.var.names is TRUE, matches the given strings exactly. If exact.var.names is FALSE, matches any variable name that begins with any element of var.names, where the * character is treated as a wild card
chains	Which chains from the MCMC to use
exact.var.names	var.names is the names of variables to include or a set of integer indices. If exact.var.names is TRUE, matches the given strings exactly. If exact.var.names is FALSE, matches any variable name that begins with any element of var.names, where the * character is treated as a wild card
rank.normalize	Whether to rank-normalize samples before calculating the R-hat, as in Vehtari 2019
sort	If TRUE, the rhats are sorted from greatest to least

is.mcmc.cache.complete *Check if a cached MCMC has completed sampling*

Description

Check if a cached MCMC has completed sampling

Usage

```
is.mcmc.cache.complete(dir)
```

Arguments

dir	The cache directory

Value

A boolean indicator of whether the cache is done sampling

mcmc.merge.parallel *Merge MCMC objects in parallel*

Description

Merges two or more mcmcsim objects by treating the chains in each as parallel chains

Usage

```
mcmc.merge.parallel(...)
```

mcmc.merge.serial *Merge MCMC objects in serial*

Description

Merges two or more mcmcsim objects by treating the chains in each a serial chains (ie, extensions of the previous chains)

Usage

```
mcmc.merge.serial(...)
```

Arguments

...	One or more mcmcsim objects or lists containing only mcmcsim objects

`pair.plot` *Generate a pairs plot of variables from an mcmc run*

Description

Generate a pairs plot of variables from an mcmc run

Usage

```
pair.plot(
  mcmc,
  var.names = mcmc@var.names,
  chains = 1:mcmc@n.chains,
  additional.burn = 0,
  additional.thin = 1,
  exact.var.names = F
)
```

Arguments

`mcmc`	A mcmc.sim object
`var.names`	var.names is the names of variables to include or a set of integer indices. If exact.var.names is TRUE, matches the given strings exactly. If exact.var.names is FALSE, matches any variable name that begins with any element of var.names, where the * character is treated as a wild card
`chains`	Which chains from the MCMC to use
`exact.var.names`	var.names is the names of variables to include or a set of integer indices. If exact.var.names is TRUE, matches the given strings exactly. If exact.var.names is FALSE, matches any variable name that begins with any element of var.names, where the * character is treated as a wild card

`process.transformations`

exporting for now for debugging

Description

exporting for now for debugging

Usage

```
process.transformations(
  var.names,
  transformations,
  n.var = length(var.names),
  var.names.name = "var.names"
)
```

---

remove.mcmc.cache *Delete previously set-up cache for running MCMC*

Description

Delete previously set-up cache for running MCMC

Usage

```
remove.mcmc.cache(dir, allow.remove.incomplete = F)
```

Arguments

dir	The directory where the cache is stored
allow.remove.incomplete	If FALSE, the function will only succeed in removing the cached files if all the MCMC chains are complete. If TRUE, removes all file regardless

---

run.mcmc.from.cache *Run MCMC from a previously set-up cache*

Description

Runs MCMC iterations from a previously set up cache

Usage

```
run.mcmc.from.cache(
  dir,
  chains = NULL,
  update.frequency = 100,
  update.detail = c("none", "low", "med", "high")[3],
  cores = parallel::detectCores(),
  remove.cache.when.done = F,
  output.file = ""
)
```

Arguments

dir	The directory where the cache is stored
chains	The numbers of the chains to run. If passed NULL (the default), runs all chains
update.frequency	How often to print updates (print an update every update.frequency iterations). If NA, no updates are printed
update.detail	The level of detail for updates
cores	The maximum number of cores to use to run chains in parallel

Value

An MCMC object with length(chains) number of chains

---

simset *The simset class*

Description

A class representing a weighted set of simulations

Slots

simulations A list of simulations

weights A numeric weight for each simulation

n.sim The number of unique simulations

parameters A matrix of parameters for each simulation, where row i corresponds to the parameters used to run the ith simulation, and each column represents a parameter value

parameter.names,n.parameters The names and number of the parameters used to run simulations

---

subset.mcmc *Subset an MCMC object*

Description

Subset an MCMC object by either burning additional iterations of the front, further thinning iterations, selecting a subset of chains, or some combination of the three

Usage

```
subset.mcmc(
  .Object,
  chains = 1:.Object@n.chains,
  additional.burn = 0,
  additional.thin = 1
)
```

Arguments

.Object	The mcmc.sim object
chains	The chains to subset
additional.burn	Additional number of iterations to burn off the front (this represents the number of iterations AFTER the thinning which has already been done to the mcmc.sim object but BEFORE applying additional.thin)
additional.thin	The factor by which to further thin iterations (applied to the already thinned simulations)

Value

An mcmc.sim object

---

subset.simset *Subset a simset*

Description

Subset a simset by indexing a subset of the included simulations

Usage

```
subset.simset(simset, indices)
```

Arguments

simset	An object of class simset
indices	Either numeric indices or a logical mask according to which to subset simulations

Value

A new simset object

thin.simset *Subset a simset by thinning its simulations*

Description

For a simset whose weights are integer values, extracts a subset of the simualtions by thinning every nth simulation (eg, if thin=3, the resulting simset keeps every third simulation). Maintains the weighting - simulations with higher weights are more likely to be kept

Usage

```
thin.simset(simset, thin)
```

Arguments

simset	A simset object
thin	How much to thin the simulations

Value

A thinned simset object

trace.plot *Generate a trace plot from an mcmc run*

Description

Generate a trace plot from an mcmc run

Usage

```
trace.plot(
  mcmc,
  var.names = mcmc@var.names,
  chains = 1:mcmc@n.chains,
  additional.burn = 0,
  additional.thin = 1,
  exact.var.names = F,
  include.rhat = mcmc@n.chains > 1,
  ncol = NULL
)
```

Arguments

mcmc	A mcmc.sim object
var.names, exact.var.names	var.names is the names of variables to include or a set of integer indices. If exact.var.names is TRUE, matches the given strings exactly. If exact.var.names is FALSE, matches any variable name that begins with any element of var.names, where the * character is treated as a wild card
chains	Which chains from the MCMC to use
include.rhat	If TRUE, includes the rhat value in the facet header
ncol	Number of columns in the facet_wrap for the resulting plot object

Index

add.parameters, 2
assemble.mcmc.from.cache, 3

cache.progress, 4
create.mcmc.cache, 4

distributions::transformation, *3*

extend.simulations, 5
extract.simset, 5
extract.simset.distribution, 6
extract.simset.parameter.distribution, 6

get.rhats, 7

is.mcmc.cache.complete, 8

mcmc.merge.parallel, 8
mcmc.merge.serial, 8
mcmc.sim, *3*, *7*, *9*, *14*

pair.plot, 9
parameter_set, *2*
process.transformations, 9

remove.mcmc.cache, 10
run.mcmc.from.cache, 10

simset, *2*, *5–7*, 11
simset-class *(*simset*)*, 11
subset.mcmc, 11
subset.simset, 12

thin.simset, 13
trace.plot, 13