

Package: mcmcr (via r-universe)

October 13, 2026

Title Manipulate MCMC Samples

Version 0.7.0.9002

Description Functions and classes to store, manipulate and summarise Monte Carlo Markov Chain (MCMC) samples. For more information see Brooks et al. (2011) <isbn:978-1-4200-7941-8>.

License MIT + file LICENSE

URL <https://github.com/poissonconsulting/mcmcr>,
<https://poissonconsulting.github.io/mcmcr/>

BugReports <https://github.com/poissonconsulting/mcmcr/issues>

Depends R (>= 4.0)

Imports abind, chk, coda, extras (>= 0.10.0), generics, lifecycle,
nlist (>= 0.5.0), purrr, stats, term, tibble, universals, utils

Suggests covr, graphics, rlang, testthat (>= 3.0.0), withr

RdMacros lifecycle

Config/Needs/website poissonconsulting/poissontemplate

Config/roxygen2/version 8.1.0

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

Repository <https://poissonconsulting.r-universe.dev>

Date/Publication 2026-10-12 21:44:16 UTC

RemoteUrl <https://github.com/poissonconsulting/mcmcr>

RemoteRef HEAD

RemoteSha dc9b3054584baebfdc1ca380c8ad322981bbd9e1

Contents

as.marray	4
as.mcmc.marray	4
as.mcmc.mcmc	5
as.mcmc.mcmcarray	6
as.mcmc.mcmcr	7
as.mcmcarray	8
as.mcmcr	8
as.mcmcrs	10
as_nlist.mcmcr	11
as_nlists.mcmcr	12
bind_chains.marray	13
bind_chains.mcmc	13
bind_chains.mcmc.list	14
bind_chains.mcmcarray	14
bind_chains.mcmcr	15
bind_dimensions	16
bind_dimensions_n	16
bind_parameters	17
check_mcmcarray	18
check_mcmcr	18
chk_mcmcr	19
coef	20
collapse_chains.mcmcr	21
combine_dimensions	22
combine_samples	23
combine_samples_n	23
converged.default	24
converged.mcmcrs	25
esr.marray	26
esr.mcmc	27
esr.mcmc.list	28
esr.mcmcarray	29
esr.mcmcr	30
esr.mcmcrs	31
ess	32
estimates.marray	32
estimates.mcmc	33
estimates.mcmc.list	34
estimates.mcmcarray	34
estimates.mcmcr	35
fill_all.marray	36
fill_all.mcmcarray	37
fill_all.mcmcr	38
fill_na.marray	39
fill_na.mcmcarray	41
fill_na.mcmcr	42

is.mccarray	43
is.mcmcarray	44
is.mcmcr	44
is.mcmcrs	45
mcmc_aperm	46
mcmc_map	46
mcmcarray-object	47
mcmcr-object	47
mcmcr_example	48
mcmcrs	48
mcmcrs-object	49
nchains.matrix	49
nchains.mccarray	50
nchains.mcmcarray	50
nchains.mcmcr	51
nchains.mcmcrs	51
niters.matrix	52
niters.mccarray	52
niters.mcmcarray	53
niters.mcmcr	54
niters.mcmcrs	54
npars.mccarray	55
npars.mcmcarray	56
npars.mcmcr	56
npdims.mcmcarray	57
npdims.mcmcr	58
nterms.mcmcarray	58
nterms.mcmcr	59
nterms.mcmcrs	59
pars.mcmcr	60
pars.mcmcrs	61
pdims.mccarray	61
pdims.mcmcarray	62
pdims.mcmcr	62
rhat.mccarray	63
rhat.mcmc	64
rhat.mcmc.list	65
rhat.mcmcarray	66
rhat.mcmcr	67
rhat.mcmcrs	68
set_pars.mcmcr	69
set_pars.mcmcrs	69
split_chains.mcmcarray	70
split_chains.mcmcr	71
subset	71
vld_mcmcr	73
zero	74

as.marray	<i>Coerce to an marray object</i>
-----------	-----------------------------------

Description

Coerces MCMC objects to an marray object.

Usage

```
as.marray(x, ...)

## S3 method for class 'list'
as.mcmcr(x, ...)
```

Arguments

x	object to coerce.
...	Unused.

Functions

- `as.mcmcr(list)`: Convert a list of uniquely named objects that can be coerced to `[mcmarray-object]`s to an mcmcr object

See Also

Other coerce: [as.mcmarray\(\)](#), [as.mcmcr\(\)](#), [mcmcrs\(\)](#)

Examples

```
as.marray(mcmcr_example$beta)
```

as.mcmc.marray	<i>Markov Chain Monte Carlo Objects</i>
----------------	---

Description

The function `mcmc` is used to create a Markov Chain Monte Carlo object. The input data are taken to be a vector, or a matrix with one column per variable.

If the optional arguments `start`, `end`, and `thin` are omitted then the chain is assumed to start with iteration 1 and have thinning interval 1. If data represents a chain that starts at a later iteration, the first iteration in the chain should be given as the `start` argument. Likewise, if data represents a chain that has already been thinned, the thinning interval should be given as the `thin` argument.

An mcmc object may be summarized by the `summary` function and visualized with the `plot` function.

MCMC objects resemble time series (ts) objects and have methods for the generic functions `time`, `start`, `end`, `frequency` and `window`.

Usage

```
## S3 method for class 'marray'
as.mcmc(x, ...)
```

Arguments

x An object that may be coerced to an mcmc object

... Further arguments to be passed to specific methods

Note

The format of the mcmc class has changed between coda version 0.3 and 0.4. Older mcmc objects will now cause `is.mcmc` to fail with an appropriate warning message. Obsolete mcmc objects can be upgraded with the `mcmcUpgrade` function.

Author(s)

Martyn Plummer

See Also

[mcmc.list](#), [mcmcUpgrade](#), [thin](#), [window.mcmc](#), [summary.mcmc](#), [plot.mcmc](#).

as.mcmc.mcmc

Markov Chain Monte Carlo Objects

Description

The function `mcmc` is used to create a Markov Chain Monte Carlo object. The input data are taken to be a vector, or a matrix with one column per variable.

If the optional arguments `start`, `end`, and `thin` are omitted then the chain is assumed to start with iteration 1 and have thinning interval 1. If data represents a chain that starts at a later iteration, the first iteration in the chain should be given as the `start` argument. Likewise, if data represents a chain that has already been thinned, the thinning interval should be given as the `thin` argument.

An mcmc object may be summarized by the `summary` function and visualized with the `plot` function.

MCMC objects resemble time series (ts) objects and have methods for the generic functions `time`, `start`, `end`, `frequency` and `window`.

Usage

```
## S3 method for class 'mcmc'
as.mcmc(x, ...)
```

Arguments

x An object that may be coerced to an mcmc object
 ... Further arguments to be passed to specific methods

Note

The format of the mcmc class has changed between coda version 0.3 and 0.4. Older mcmc objects will now cause `is.mcmc` to fail with an appropriate warning message. Obsolete mcmc objects can be upgraded with the `mcmcUpgrade` function.

Author(s)

Martyn Plummer

See Also

[mcmc.list](#), [mcmcUpgrade](#), [thin](#), [window.mcmc](#), [summary.mcmc](#), [plot.mcmc](#).

as.mcmc.mcmcarray

Markov Chain Monte Carlo Objects

Description

The function `mcmc` is used to create a Markov Chain Monte Carlo object. The input data are taken to be a vector, or a matrix with one column per variable.

If the optional arguments `start`, `end`, and `thin` are omitted then the chain is assumed to start with iteration 1 and have thinning interval 1. If data represents a chain that starts at a later iteration, the first iteration in the chain should be given as the `start` argument. Likewise, if data represents a chain that has already been thinned, the thinning interval should be given as the `thin` argument.

An mcmc object may be summarized by the `summary` function and visualized with the `plot` function.

MCMC objects resemble time series (ts) objects and have methods for the generic functions `time`, `start`, `end`, `frequency` and `window`.

Usage

```
## S3 method for class 'mcmcarray'
as.mcmc(x, ...)
```

Arguments

x An object that may be coerced to an mcmc object
 ... Further arguments to be passed to specific methods

Note

The format of the mcmc class has changed between coda version 0.3 and 0.4. Older mcmc objects will now cause `is.mcmc` to fail with an appropriate warning message. Obsolete mcmc objects can be upgraded with the `mcmcUpgrade` function.

Author(s)

Martyn Plummer

See Also

[mcmc.list](#), [mcmcUpgrade](#), [thin](#), [window.mcmc](#), [summary.mcmc](#), [plot.mcmc](#).

as.mcmc.mcmc

Markov Chain Monte Carlo Objects

Description

The function `mcmc` is used to create a Markov Chain Monte Carlo object. The input data are taken to be a vector, or a matrix with one column per variable.

If the optional arguments `start`, `end`, and `thin` are omitted then the chain is assumed to start with iteration 1 and have thinning interval 1. If data represents a chain that starts at a later iteration, the first iteration in the chain should be given as the `start` argument. Likewise, if data represents a chain that has already been thinned, the thinning interval should be given as the `thin` argument.

An mcmc object may be summarized by the `summary` function and visualized with the `plot` function.

MCMC objects resemble time series (ts) objects and have methods for the generic functions `time`, `start`, `end`, `frequency` and `window`.

Usage

```
## S3 method for class 'mcmc'
as.mcmc(x, ...)
```

Arguments

<code>x</code>	An object that may be coerced to an mcmc object
<code>...</code>	Further arguments to be passed to specific methods

Note

The format of the mcmc class has changed between coda version 0.3 and 0.4. Older mcmc objects will now cause `is.mcmc` to fail with an appropriate warning message. Obsolete mcmc objects can be upgraded with the `mcmcUpgrade` function.

Author(s)

Martyn Plummer

See Also

[mcmc.list](#), [mcmcUpgrade](#), [thin](#), [window.mcmc](#), [summary.mcmc](#), [plot.mcmc](#).

as.mcmcarray

Coerce to an mcmcarray object

Description

Coerces MCMC objects to an [mcmcarray-object\(\)](#).

Usage

```
as.mcmcarray(x, ...)
```

Arguments

x	object to coerce.
...	Unused.

See Also

Other coerce: [as.marray\(\)](#), [as.mcmc\(\)](#), [mcmcrs\(\)](#)

Examples

```
as.mcmcarray(as.marray(mcmcr_example$beta))
```

as.mcmcr

Convert to an mcmcr object

Description

Converts an MCMC object to an [mcmcr-object\(\)](#).

Usage

```

as.mcmcr(x, ...)

## S3 method for class 'marray'
as.mcmcr(x, name = "par", ...)

## S3 method for class 'mcmarray'
as.mcmcr(x, name = "par", ...)

## S3 method for class 'nlist'
as.mcmcr(x, ...)

## S3 method for class 'nlists'
as.mcmcr(x, ...)

## S3 method for class 'mcmc'
as.mcmcr(x, ...)

## S3 method for class 'mcmc.list'
as.mcmcr(x, ...)

## S3 method for class 'mcmcrs'
as.mcmcr(x, ...)

```

Arguments

x	An MCMC object.
...	Unused.
name	A string specifying the parameter name.

Value

An mcmcr object.

Methods (by class)

- `as.mcmcr(marray)`: Convert an `marray` object to an `mcmcr` object
- `as.mcmcr(mcmarray)`: Convert an `mcmarray-object()` to an `mcmcr` object
- `as.mcmcr(nlist)`: Convert an `nlist::nlist-object()` to an `mcmcr` object
- `as.mcmcr(nlists)`: Convert an `nlist::nlists-object()` to an `mcmcr` object
- `as.mcmcr(mcmc)`: Convert an `coda::mcmc()` object to an `mcmcr` object
- `as.mcmcr(mcmc.list)`: Convert an `coda::mcmc.list()` object to an `mcmcr` object
- `as.mcmcr(mcmcrs)`: Convert an `mcmcrs-object()` to an `mcmcr` object

See Also

Other coerce: `as.marray()`, `as.mcmarray()`, `mcmcrs()`

Examples

```
mcmc.list <- coda::as.mcmc.list(mcmc::mcmc_example)
as.mcmcrc(mcmc.list)
```

as.mcmcrcs	<i>Convert to an mcmcrcs object</i>
------------	-------------------------------------

Description

Converts an MCMC object to an [mcmcrcs-object\(\)](#).

Usage

```
as.mcmcrcs(x, ...)

## S3 method for class 'list'
as.mcmcrcs(x, ...)

## S3 method for class 'mcmc'
as.mcmcrcs(x, name = "mcmc1", ...)
```

Arguments

x	An MCMC object.
...	Unused.
name	A string specifying the element name.

Value

An mcmcrcs object.

Methods (by class)

- `as.mcmcrcs(list)`: Convert a list of `[mcmcrcs-object]`s to an mcmcrcs object
- `as.mcmcrcs(mcmc)`: Convert an [mcmcrcs-object\(\)](#) to an mcmcrcs object

Examples

```
as.mcmcrcs(mcmc::mcmc_example)
```

as_nlist.mcmc	<i>Coerce to nlist</i>
---------------	------------------------

Description

Coerce an R object to an [nlist_object\(\)](#).

Usage

```
## S3 method for class 'mcmc'  
as_nlist(x, ...)
```

Arguments

x	An object.
...	Unused.

Value

An nlist object.

Methods (by class)

- `as_nlist(numeric)`: Coerce named numeric vector to nlist
- `as_nlist(list)`: Coerce list to nlist
- `as_nlist(data.frame)`: Coerce data.frame to nlist
- `as_nlist(mcmc)`: Coerce mcmc (with one iteration) to nlist
- `as_nlist(mcmc.list)`: Coerce mcmc.list (with one iteration) to nlist

See Also

Other coerce: [as_nlists\(\)](#)

Examples

```
as_nlist(list(x = 1:4))  
as_nlist(c(`a[2]` = 3, `a[1]` = 2))
```

as_nlists.mcmc	<i>Coerce to nlists</i>
----------------	-------------------------

Description

Coerce an R object to an [nlists_object\(\)](#).

Usage

```
## S3 method for class 'mcmc'  
as_nlists(x, ...)
```

Arguments

x	An object.
...	Unused.

Value

An nlists object.

Methods (by class)

- `as_nlists(list)`: Coerce list to nlists
- `as_nlists(mcmc)`: Coerce mcmc to nlists
- `as_nlists(mcmc.list)`: Coerce mcmc.list to nlists
- `as_nlists(nlist)`: Coerce nlist to nlists

See Also

Other coerce: [as_nlist\(\)](#)

Examples

```
as_nlists(list(nlist(x = c(1, 5)), nlist(x = c(2, 3)), nlist(x = c(3, 2))))
```

bind_chains.marray *Bind by Chains.*

Description

Binds two MCMC objects (with the same parameters and iterations) by chains.

Usage

```
## S3 method for class 'marray'
bind_chains(x, x2, ...)
```

Arguments

x	An object.
x2	A second object.
...	Other arguments passed to methods.

Value

The combined object.

See Also

Other MCMC manipulations: [bind_iterations\(\)](#), [collapse_chains\(\)](#), [estimates\(\)](#), [split_chains\(\)](#)

bind_chains.mcmc *Bind by Chains.*

Description

Binds two MCMC objects (with the same parameters and iterations) by chains.

Usage

```
## S3 method for class 'mcmc'
bind_chains(x, x2, ...)
```

Arguments

x	An object.
x2	A second object.
...	Other arguments passed to methods.

Value

The combined object.

See Also

Other MCMC manipulations: [bind_iterations\(\)](#), [collapse_chains\(\)](#), [estimates\(\)](#), [split_chains\(\)](#)

`bind_chains.mcmc.list` *Bind by Chains.*

Description

Binds two MCMC objects (with the same parameters and iterations) by chains.

Usage

```
## S3 method for class 'mcmc.list'
bind_chains(x, x2, ...)
```

Arguments

<code>x</code>	An object.
<code>x2</code>	A second object.
<code>...</code>	Other arguments passed to methods.

Value

The combined object.

See Also

Other MCMC manipulations: [bind_iterations\(\)](#), [collapse_chains\(\)](#), [estimates\(\)](#), [split_chains\(\)](#)

`bind_chains.mcmcarray` *Bind by Chains.*

Description

Binds two MCMC objects (with the same parameters and iterations) by chains.

Usage

```
## S3 method for class 'mcmcarray'
bind_chains(x, x2, ...)
```

Arguments

x	An object.
x2	A second object.
...	Other arguments passed to methods.

Value

The combined object.

See Also

Other MCMC manipulations: [bind_iterations\(\)](#), [collapse_chains\(\)](#), [estimates\(\)](#), [split_chains\(\)](#)

bind_chains.mcmc	<i>Bind by Chains.</i>
------------------	------------------------

Description

Binds two MCMC objects (with the same parameters and iterations) by chains.

Usage

```
## S3 method for class 'mcmc'  
bind_chains(x, x2, ...)
```

Arguments

x	An object.
x2	A second object.
...	Other arguments passed to methods.

Value

The combined object.

See Also

Other MCMC manipulations: [bind_iterations\(\)](#), [collapse_chains\(\)](#), [estimates\(\)](#), [split_chains\(\)](#)

bind_dimensions	<i>Combine two MCMC objects by dimensions</i>
-----------------	---

Description

Combines multiple MCMC objects (with the same parameters, chains and iterations) by parameter dimensions.

Usage

```
bind_dimensions(x, x2, along = NULL, ...)
```

Arguments

x	An MCMC object.
x2	a second MCMC object.
along	A count (or NULL) indicating the parameter dimension to bind along. For an mcmcr-object() it may also be a vector with one value per parameter, in the order the parameters occur in x.
...	Unused.

See Also

[universals::bind_chains\(\)](#)

Other bind: [bind_dimensions_n\(\)](#), [bind_parameters\(\)](#)

Examples

```
bind_dimensions(mcmcr_example, mcmcr_example)
```

bind_dimensions_n	<i>Combine multiple MCMC objects by parameter dimensions</i>
-------------------	--

Description

Combines multiple MCMC objects (with the same parameters, chains and iterations) by parameter dimensions.

Usage

```
bind_dimensions_n(...)
```

Arguments

...	one or more MCMC objects
-----	--------------------------

See Also

[universals::bind_chains\(\)](#)

Other bind: [bind_dimensions\(\)](#), [bind_parameters\(\)](#)

Examples

```
bind_dimensions_n(mcmcr_example, mcmcr_example, mcmcr_example)
```

bind_parameters	<i>Combine two MCMC object by parameters</i>
-----------------	--

Description

Combines two MCMC objects (with the same chains and iterations) by their parameters.

Usage

```
bind_parameters(x, x2, ...)
```

Arguments

x	an MCMC object
x2	a second MCMC object.
...	Unused.

See Also

[universals::bind_chains\(\)](#)

Other bind: [bind_dimensions\(\)](#), [bind_dimensions_n\(\)](#)

Examples

```
bind_parameters(  
  subset(mcmcr_example, pars = "sigma"),  
  subset(mcmcr_example, pars = "beta")  
)
```

check_mcmcarray	<i>Check mcmcarray</i> [Deprecated]
-----------------	--

Description

Check mcmcarray **[Deprecated]**

Usage

```
check_mcmcarray(x, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (it if doesn't throw an error).

See Also

[check_mcmc\(\)](#)

Examples

```
check_mcmcarray(mcmc::mcmc_example$beta)
```

check_mcmc	<i>Check mcmc</i> [Deprecated]
------------	---------------------------------------

Description

Check mcmc **[Deprecated]**

Usage

```
check_mcmc(x, sorted = FALSE, x_name = substitute(x), error = TRUE)
```

Arguments

x	The object to check.
sorted	A flag specifying whether the parameters must be sorted.
x_name	A string of the name of the object.
error	A flag indicating whether to throw an informative error or immediately generate an informative message if the check fails.

Value

An invisible copy of x (it if doesn't throw an error).

See Also

[check_mcmc\(\)](#)

Examples

```
check_mcmc(mcmc::mcmc_example)
```

chk_mcmc	<i>Check MCMC objects</i>
----------	---------------------------

Description

Checks class and structure of MCMC objects.

chk_mcmcarray checks if [mcmcarray-object\(\)](#) object using `is.array(x) && is.numeric(x)`

chk_mcmc checks if an [mcmc-object\(\)](#).

chk_mmcrcs checks if an [mcrcs-object\(\)](#).

Usage

```
chk_mcmcarray(x, x_name = NULL)
```

```
chk_mcmc(x, x_name = NULL)
```

```
chk_mmcrcs(x, x_name = NULL)
```

Arguments

x	The object to check.
x_name	A string of the name of object x or NULL.

Details

To just check class use [chk::chk_s3_class\(\)](#).

Value

NULL, invisibly. Called for the side effect of throwing an error if the condition is not met.

Functions

- `chk_mcmcarray()`: Check mcmcarray Object
- `chk_mcmcr()`: Check mcmcr Object
- `chk_mmcrcs()`: Check mcrcs Object

See Also

[vld_mcmcr\(\)](#)

Examples

```
# chk_mcmcarray
try(chk_mcmcarray(1))

# chk_mcmcr
chk_mcmcr(as.mcmcr(list(x = 1)))
try(chk_mcmcr(1))

# chk_mmcrcs
chk_mmcrcs(as.mmcrcs(as.mcmcr(list(x = 1))))
try(chk_mmcrcs(1))
```

coef

Term coefficients

Description

Gets coefficients for all the terms in an MCMC object.

Usage

```
## S3 method for class 'mcmc'
coef(
  object,
  conf_level = 0.95,
  estimate = median,
  simplify = TRUE,
  directional_information = FALSE,
  ...
)
```

Arguments

object	The MCMC object to get the coefficients for
conf_level	A number specifying the confidence level. By default 0.95.
estimate	The function to use to calculate the estimate.
simplify	[Deprecated] Must be TRUE.
directional_information	A flag specifying whether the svalue column should be calculated using <code>extras::directional_information()</code> instead of <code>extras::svalue()</code> . The default value will change from FALSE to TRUE in a future release; set the argument explicitly to avoid the deprecation warning.
...	Unused.

Value

An data frame of the coefficients with the columns indicating the term, estimate, lower and upper credible intervals and svalue

Methods (by class)

- `coef(mcmc)`: Get coefficients for terms in mcmc object

See Also

`stats::coef`

Examples

```
coef(mcmc_example, directional_information = FALSE)
coef(mcmc_example, directional_information = TRUE)
```

collapse_chains.mcmc *Collapse Chains*

Description

Collapses an MCMC object's chains into a single chain.

Usage

```
## S3 method for class 'mcmc'
collapse_chains(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

The modified object with one chain.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [estimates\(\)](#), [split_chains\(\)](#)

combine_dimensions	<i>Combine samples by dimensions</i>
--------------------	--------------------------------------

Description

Combines MCMC object samples by dimensions along along using fun.

Usage

```
combine_dimensions(x, fun = mean, along = NULL, ...)
```

Arguments

x	An MCMC object
fun	The function to use when combining dimensions
along	A positive integer (or NULL) indicating the parameter dimension(s) to bind along.
...	Unused.

Value

The MCMC object with reduced dimensions.

See Also

Other combine: [combine_samples\(\)](#), [combine_samples_n\(\)](#)

Examples

```
combine_dimensions(mcmcr_example$alpha)
```

combine_samples	<i>Combine MCMC samples of two objects</i>
-----------------	--

Description

Combines samples of two MCMC objects (with the same parameters, chains and iterations) using a function.

Usage

```
combine_samples(x, x2, fun = mean, ...)
```

Arguments

x	An MCMC object.
x2	a second MCMC object.
fun	The function to use to combine the samples. The function must return a scalar.
...	Unused.

Value

The combined samples as an MCMC object with the same parameters, chains and iterations as the original objects.

See Also

Other combine: [combine_dimensions\(\)](#), [combine_samples_n\(\)](#)

Examples

```
combine_samples(mcmcr_example, mcmcr_example, fun = sum)
```

combine_samples_n	<i>Combine MCMC samples of multiple objects</i>
-------------------	---

Description

Combines samples of multiple MCMC objects (with the same parameters, chains and iterations) using a function.

Usage

```
combine_samples_n(x, ..., fun = mean)
```

Arguments

<code>x</code>	An MCMC object (or a list of mcmc objects).
<code>...</code>	Additional MCMC objects.
<code>fun</code>	A function.

See Also

Other combine: [combine_dimensions\(\)](#), [combine_samples\(\)](#)

Examples

```
combine_samples_n(mcmcr_example, mcmcr_example, mcmcr_example, fun = sum)
```

<code>converged.default</code>	<i>Converged</i>
--------------------------------	------------------

Description

Tests whether an object has converged.

Usage

```
## Default S3 method:
converged(
  x,
  rhat = 1.1,
  esr = 0.33,
  by = "all",
  as_df = FALSE,
  na_rm = FALSE,
  ...
)
```

Arguments

<code>x</code>	An object.
<code>rhat</code>	The maximum rhat value.
<code>esr</code>	The minimum effective sampling rate.
<code>by</code>	A string indicating whether to determine by "term", "parameter" or "all".
<code>as_df</code>	A flag indicating whether to return the results as a data frame versus a named list.
<code>na_rm</code>	A flag specifying whether to ignore missing values.
<code>...</code>	Other arguments passed to methods.

Value

A logical scalar indicating whether the object has converged.

See Also

Other convergence: [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

Examples

```
converged(mcmcr_example)
```

converged.mcmcrcs	<i>Converged</i>
-------------------	------------------

Description

Tests whether an object has converged.

Usage

```
## S3 method for class 'mcmcrcs'
converged(
  x,
  rhat = 1.1,
  esr = 0.33,
  by = "all",
  as_df = FALSE,
  bound = FALSE,
  na_rm = FALSE,
  ...
)
```

Arguments

x	An object.
rhat	The maximum rhat value.
esr	The minimum effective sampling rate.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
bound	flag specifying whether to bind mcmcrcs objects by their chains before calculating rhat.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Value

A logical scalar indicating whether the object has converged.

See Also

Other convergence: [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

Examples

```
converged(mcmcrs(mcmcr_example, mcmcr_example))
converged(mcmcrs(mcmcr_example, mcmcr_example), bound = TRUE)
```

esr.marray	<i>Effective Sampling Rate</i>
------------	--------------------------------

Description

Calculates the effective sampling rate (esr).

Usage

```
## S3 method for class 'marray'
esr(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default

$$\frac{1}{1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta)}$$

from Brooks et al. (2011) where the infinite sum is truncated at lag k when $\rho_{k+1}(\theta) < 0$.

Value

A number between 0 and 1 indicating the esr value.

References

Brooks, S., Gelman, A., Jones, G.L., and Meng, X.-L. (Editors). 2011. Handbook for Markov Chain Monte Carlo. Taylor & Francis, Boca Raton.

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

esr.mcmc	<i>Effective Sampling Rate</i>
----------	--------------------------------

Description

Calculates the effective sampling rate (esr).

Usage

```
## S3 method for class 'mcmc'
esr(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default

$$\frac{1}{1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta)}$$

from Brooks et al. (2011) where the infinite sum is truncated at lag k when $\rho_{k+1}(\theta) < 0$.

Value

A number between 0 and 1 indicating the esr value.

References

Brooks, S., Gelman, A., Jones, G.L., and Meng, X.-L. (Editors). 2011. Handbook for Markov Chain Monte Carlo. Taylor & Francis, Boca Raton.

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

esr.mcmc.list

Effective Sampling Rate

Description

Calculates the effective sampling rate (esr).

Usage

```
## S3 method for class 'mcmc.list'
esr(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default

$$\frac{1}{1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta)}$$

from Brooks et al. (2011) where the infinite sum is truncated at lag k when $\rho_{k+1}(\theta) < 0$.

Value

A number between 0 and 1 indicating the esr value.

References

Brooks, S., Gelman, A., Jones, G.L., and Meng, X.-L. (Editors). 2011. Handbook for Markov Chain Monte Carlo. Taylor & Francis, Boca Raton.

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

esr.mcmcarray

*Effective Sampling Rate***Description**

Calculates the effective sampling rate (esr).

Usage

```
## S3 method for class 'mcmcarray'
esr(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default

$$\frac{1}{1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta)}$$

from Brooks et al. (2011) where the infinite sum is truncated at lag k when $\rho_{k+1}(\theta) < 0$.

Value

A number between 0 and 1 indicating the esr value.

References

Brooks, S., Gelman, A., Jones, G.L., and Meng, X.-L. (Editors). 2011. Handbook for Markov Chain Monte Carlo. Taylor & Francis, Boca Raton.

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

esr.mcmc

Effective Sampling Rate

Description

Calculates the effective sampling rate (esr).

Usage

```
## S3 method for class 'mcmc'
esr(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default

$$\frac{1}{1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta)}$$

from Brooks et al. (2011) where the infinite sum is truncated at lag k when $\rho_{k+1}(\theta) < 0$.

Value

A number between 0 and 1 indicating the esr value.

References

Brooks, S., Gelman, A., Jones, G.L., and Meng, X.-L. (Editors). 2011. Handbook for Markov Chain Monte Carlo. Taylor & Francis, Boca Raton.

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

Examples

```
esr(mcmc_example)
```

esr.mcmcrcs	<i>Effective Sampling Rate</i>
-------------	--------------------------------

Description

Calculates the effective sampling rate (esr).

Usage

```
## S3 method for class 'mcmcrcs'
esr(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default

$$\frac{1}{1 + 2 \sum_{k=1}^{\infty} \rho_k(\theta)}$$

from Brooks et al. (2011) where the infinite sum is truncated at lag k when $\rho_{k+1}(\theta) < 0$.

Value

A number between 0 and 1 indicating the esr value.

References

Brooks, S., Gelman, A., Jones, G.L., and Meng, X.-L. (Editors). 2011. Handbook for Markov Chain Monte Carlo. Taylor & Francis, Boca Raton.

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

Examples

```
esr(mcmcrcs(mcmcr_example, mcmcr_example))
```

ess	<i>Effective Sample Size</i>
-----	------------------------------

Description

Calculates the effective sample size based on [esr\(\)](#).

Usage

```
ess(x, by = "all", as_df = FALSE)
```

Arguments

x	An MCMC object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.

See Also

[universals::esr](#)

Examples

```
ess(mcmcr_example)
```

estimates.marray	<i>Estimates</i>
------------------	------------------

Description

Calculates the estimates for an MCMC object.

Usage

```
## S3 method for class 'marray'
estimates(x, fun = median, as_df = FALSE, ...)
```

Arguments

x	An object.
fun	A function that given a numeric vector returns a numeric scalar.
as_df	A flag indicating whether to return the results as a data frame versus a named list.
...	Optional arguments to fun.

Value

A named list or data frame.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [collapse_chains\(\)](#), [split_chains\(\)](#)

estimates.mcmc	<i>Estimates</i>
----------------	------------------

Description

Calculates the estimates for an MCMC object.

Usage

```
## S3 method for class 'mcmc'  
estimates(x, fun = median, as_df = FALSE, ...)
```

Arguments

x	An object.
fun	A function that given a numeric vector returns a numeric scalar.
as_df	A flag indicating whether to return the results as a data frame versus a named list.
...	Optional arguments to fun.

Value

A named list or data frame.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [collapse_chains\(\)](#), [split_chains\(\)](#)

`estimates.mcmc.list` *Estimates*

Description

Calculates the estimates for an MCMC object.

Usage

```
## S3 method for class 'mcmc.list'
estimates(x, fun = median, as_df = FALSE, ...)
```

Arguments

<code>x</code>	An object.
<code>fun</code>	A function that given a numeric vector returns a numeric scalar.
<code>as_df</code>	A flag indicating whether to return the results as a data frame versus a named list.
<code>...</code>	Optional arguments to <code>fun</code> .

Value

A named list or data frame.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [collapse_chains\(\)](#), [split_chains\(\)](#)

`estimates.mcmcarray` *Estimates*

Description

Calculates the estimates for an MCMC object.

Usage

```
## S3 method for class 'mcmcarray'
estimates(x, fun = median, as_df = FALSE, ...)
```

Arguments

<code>x</code>	An object.
<code>fun</code>	A function that given a numeric vector returns a numeric scalar.
<code>as_df</code>	A flag indicating whether to return the results as a data frame versus a named list.
<code>...</code>	Optional arguments to <code>fun</code> .

Value

A named list or data frame.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [collapse_chains\(\)](#), [split_chains\(\)](#)

estimates.mcmc	<i>Estimates</i>
----------------	------------------

Description

Calculates the estimates for an MCMC object.

Usage

```
## S3 method for class 'mcmc'  
estimates(x, fun = median, as_df = FALSE, ...)
```

Arguments

x	An object.
fun	A function that given a numeric vector returns a numeric scalar.
as_df	A flag indicating whether to return the results as a data frame versus a named list.
...	Optional arguments to fun.

Value

A named list or data frame.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [collapse_chains\(\)](#), [split_chains\(\)](#)

Examples

```
estimates(mcmc_example)
```

fill_all.marray	<i>Fill All Values</i>
-----------------	------------------------

Description

Fills all of an object's (missing and non-missing) values while preserving the object's dimensionality and class.

Usage

```
## S3 method for class 'marray'  
fill_all(x, value = 0, nas = TRUE, ...)
```

Arguments

x	An object.
value	A scalar of the value to replace values with.
nas	A flag specifying whether to also fill missing values.
...	Other arguments passed to methods.

Details

It should only be defined for objects with values of consistent class ie not standard data.frames.

Value

The modified object.

Methods (by class)

- `fill_all(logical)`: Fill All for logical Objects
- `fill_all(integer)`: Fill All for integer Objects
- `fill_all(numeric)`: Fill All for numeric Objects
- `fill_all(character)`: Fill All for character Objects

See Also

Other fill: [fill_na\(\)](#)

Examples

```
# logical
fill_all(c(TRUE, NA, FALSE))
fill_all(c(TRUE, NA, FALSE, nas = FALSE))
fill_all(c(TRUE, NA, FALSE, value = NA))

# integer
fill_all(matrix(1:4, nrow = 2), value = -1)

# numeric
fill_all(c(1, 4, NA), value = TRUE)
fill_all(c(1, 4, NA), value = TRUE, nas = FALSE)

# character
fill_all(c("some", "words"), value = TRUE)
```

fill_all.mcmcarray	<i>Fill All Values</i>
--------------------	------------------------

Description

Fills all of an object's (missing and non-missing) values while preserving the object's dimensionality and class.

Usage

```
## S3 method for class 'mcmcarray'
fill_all(x, value = 0, nas = TRUE, ...)
```

Arguments

x	An object.
value	A scalar of the value to replace values with.
nas	A flag specifying whether to also fill missing values.
...	Other arguments passed to methods.

Details

It should only be defined for objects with values of consistent class ie not standard data.frames.

Value

The modified object.

Methods (by class)

- `fill_all(logical)`: Fill All for logical Objects
- `fill_all(integer)`: Fill All for integer Objects
- `fill_all(numeric)`: Fill All for numeric Objects
- `fill_all(character)`: Fill All for character Objects

See Also

Other fill: [fill_na\(\)](#)

Examples

```
# logical
fill_all(c(TRUE, NA, FALSE))
fill_all(c(TRUE, NA, FALSE, nas = FALSE))
fill_all(c(TRUE, NA, FALSE, value = NA))

# integer
fill_all(matrix(1:4, nrow = 2), value = -1)

# numeric
fill_all(c(1, 4, NA), value = TRUE)
fill_all(c(1, 4, NA), value = TRUE, nas = FALSE)

# character
fill_all(c("some", "words"), value = TRUE)
```

fill_all.mcmc

Fill All Values

Description

Fills all of an object's (missing and non-missing) values while preserving the object's dimensionality and class.

Usage

```
## S3 method for class 'mcmc'
fill_all(x, value = 0, nas = TRUE, ...)
```

Arguments

<code>x</code>	An object.
<code>value</code>	A scalar of the value to replace values with.
<code>nas</code>	A flag specifying whether to also fill missing values.
<code>...</code>	Other arguments passed to methods.

Details

It should only be defined for objects with values of consistent class ie not standard data.frames.

Value

The modified object.

Methods (by class)

- `fill_all(logical)`: Fill All for logical Objects
- `fill_all(integer)`: Fill All for integer Objects
- `fill_all(numeric)`: Fill All for numeric Objects
- `fill_all(character)`: Fill All for character Objects

See Also

Other fill: [fill_na\(\)](#)

Examples

```
# logical
fill_all(c(TRUE, NA, FALSE))
fill_all(c(TRUE, NA, FALSE, nas = FALSE))
fill_all(c(TRUE, NA, FALSE, value = NA))

# integer
fill_all(matrix(1:4, nrow = 2), value = -1)

# numeric
fill_all(c(1, 4, NA), value = TRUE)
fill_all(c(1, 4, NA), value = TRUE, nas = FALSE)

# character
fill_all(c("some", "words"), value = TRUE)
```

fill_na.marray

Fill Missing Values

Description

Fills all of an object's missing values while preserving the object's dimensionality and class.

Usage

```
## S3 method for class 'marray'
fill_na(x, value = 0, ...)
```

Arguments

x	An object.
value	A scalar of the value to replace values with.
...	Other arguments passed to methods.

Details

It should only be defined for objects with values of consistent class ie not standard data.frames.

Value

The modified object.

Methods (by class)

- fill_na(logical): Fill Missing Values for logical Objects
- fill_na(integer): Fill Missing Values for integer Objects
- fill_na(numeric): Fill Missing Values for numeric Objects
- fill_na(character): Fill Missing Values for character Objects

See Also

Other fill: [fill_all\(\)](#)

Examples

```
# logical
fill_na(c(TRUE, NA))

# integer
fill_na(c(1L, NA), 0)

# numeric
fill_na(c(1, NA), Inf)

# character
fill_na(c("text", NA))
fill_na(matrix(c("text", NA)), value = Inf)
```

fill_na.mcmcarray	<i>Fill Missing Values</i>
-------------------	----------------------------

Description

Fills all of an object's missing values while preserving the object's dimensionality and class.

Usage

```
## S3 method for class 'mcmcarray'  
fill_na(x, value = 0, ...)
```

Arguments

x	An object.
value	A scalar of the value to replace values with.
...	Other arguments passed to methods.

Details

It should only be defined for objects with values of consistent class ie not standard data.frames.

Value

The modified object.

Methods (by class)

- fill_na(logical): Fill Missing Values for logical Objects
- fill_na(integer): Fill Missing Values for integer Objects
- fill_na(numeric): Fill Missing Values for numeric Objects
- fill_na(character): Fill Missing Values for character Objects

See Also

Other fill: [fill_all\(\)](#)

Examples

```
# logical  
fill_na(c(TRUE, NA))  
  
# integer  
fill_na(c(1L, NA), 0)  
  
# numeric  
fill_na(c(1, NA), Inf)
```

```
# character
fill_na(c("text", NA))
fill_na(matrix(c("text", NA)), value = Inf)
```

fill_na.mcmc

Fill Missing Values

Description

Fills all of an object's missing values while preserving the object's dimensionality and class.

Usage

```
## S3 method for class 'mcmc'
fill_na(x, value = 0, ...)
```

Arguments

x	An object.
value	A scalar of the value to replace values with.
...	Other arguments passed to methods.

Details

It should only be defined for objects with values of consistent class ie not standard data.frames.

Value

The modified object.

Methods (by class)

- fill_na(logical): Fill Missing Values for logical Objects
- fill_na(integer): Fill Missing Values for integer Objects
- fill_na(numeric): Fill Missing Values for numeric Objects
- fill_na(character): Fill Missing Values for character Objects

See Also

Other fill: [fill_all\(\)](#)

Examples

```
# logical
fill_na(c(TRUE, NA))

# integer
fill_na(c(1L, NA), 0)

# numeric
fill_na(c(1, NA), Inf)

# character
fill_na(c("text", NA))
fill_na(matrix(c("text", NA)), value = Inf)
```

*is.marray**Is marray object*

Description

Tests whether an object is an marray.

Usage

```
is.marray(x)
```

Arguments

x The object to test.

Value

A flag indicating whether the test was positive.

See Also

Other is: [is.marray\(\)](#), [is.marray\(\)](#), [is.marray\(\)](#)

Examples

```
is.marray(marray_example)
```

is.mcmarray	<i>Is mcmarray object</i>
-------------	---------------------------

Description

Tests whether an object is an `mcmarray-object()`.

Usage

```
is.mcmarray(x)
```

Arguments

x	The object to test.
---	---------------------

Value

A flag indicating whether the test was positive.

See Also

Other is: `is.marray()`, `is.mcmcr()`, `is.mcmcrs()`

Examples

```
is.mcmarray(mcmcr_example$beta)
```

is.mcmcr	<i>Is mcmcr object</i>
----------	------------------------

Description

Tests whether an object is an `mcmcr-object()`.

Usage

```
is.mcmcr(x)
```

Arguments

x	The object to test.
---	---------------------

Value

A flag indicating whether the test was positive.

See Also

Other is: [is.marray\(\)](#), [is.mmcarray\(\)](#), [is.mcmcrs\(\)](#)

Examples

```
is.mcmcr(mcmcr_example)
```

is.mcmcrs

Is mcmcrs object

Description

Tests whether an object is an [mcmcrs-object\(\)](#).

Usage

```
is.mcmcrs(x)
```

Arguments

x The object to test.

Value

A flag indicating whether the test was positive.

See Also

Other is: [is.marray\(\)](#), [is.mmcarray\(\)](#), [is.mcmcr\(\)](#)

Examples

```
is.mcmcrs(mcmcrs(mcmcr_example))
```

mcmc_aperm	<i>MCMC object transposition</i>
------------	----------------------------------

Description

Transpose an MCMC object by permuting its parameter dimensions.

Usage

```
mcmc_aperm(x, perm, ...)
```

Arguments

x	The MCMC object to transpose. Missing parameter dimensions are added on the end. If perm = NULL (the default) the parameter dimensions are reversed.
perm	A integer vector of the new order for the parameter dimensions.
...	Unused.

Value

The modified MCMC object

See Also

Other manipulate: [mcmc_map\(\)](#)

mcmc_map	<i>MCMC map</i>
----------	-----------------

Description

Adjust the sample values of an MCMC object using a function.

Usage

```
mcmc_map(.x, .f, .by = 1:npdims(.x), ...)
```

Arguments

.x	An MCMC object
.f	The function to use
.by	A positive integer vector of the dimensions to apply the function over.
...	Additional arguments passed to .f.

Value

The updated MCMC object.

See Also

Other manipulate: [mcmc_aperm\(\)](#)

Examples

```
mcmc_map(mcmcr_example$beta, exp)
```

mcmcarray-object	<i>mcmcarray</i>
------------------	------------------

Description

An mcmcarray object is an array where the first dimension is the chains, the second dimension is the iterations and the subsequent dimensions represent the dimensionality of the parameter. The name mcmcarray reflects the fact that the MCMC dimensions, ie the chains and iterations, precede the parameter dimensions.

See Also

Other objects: [mcmcr-object](#), [mcmcrs-object](#)

Examples

```
mcmcr_example$beta
```

mcmcr-object	<i>mcmcr</i>
--------------	--------------

Description

An mcmcr object stores multiple uniquely named [mcmcarray-object\(\)](#) objects with the same number of chains and iterations.

Details

mcmcr objects allow a set of dimensionality preserving parameters to be manipulated and queried as a whole.

See Also

Other objects: [mcmcarray-object](#), [mcmcrs-object](#)

Examples

```
mcmcr_example
```

mcmcr_example	<i>An example mcmcr object</i>
---------------	--------------------------------

Description

An example `mcmcr-object()` derived from `coda::line()`.

Usage

```
mcmcr_example
```

Format

An object of class mcmcr of length 3.

Examples

```
mcmcr_example
```

mcmcrs	<i>Create mcmcrs</i>
--------	----------------------

Description

Creates an `mcmcrs-object()` from multiple `link{mcmcr-object}`s.

Usage

```
mcmcrs(...)
```

Arguments

... Objects of class mcmcr.

Value

An object of class mcmcrs

See Also

Other coerce: `as.marray()`, `as.mcmarray()`, `as.mcmcr()`

Examples

```
mcmcrs(mcmcr_example, mcmcr_example)
```

mcmcrrs-object	<i>mcmcrrs</i>
----------------	----------------

Description

An `mcmcrrs` object stores multiple `mcmcr-object()`s with the same parameters and the same number of chains and iterations.

Details

`mcmcrrs` objects allow the results of multiple analyses using the same model to be manipulated and queried as a whole.

See Also

Other objects: `mcmcarray-object`, `mcmcr-object`

Examples

```
mcmcrrs(mcmcr_example, mcmcr_example)
```

<code>nchains.matrix</code>	<i>Number of Chains</i>
-----------------------------	-------------------------

Description

Gets the number of chains of an MCMC object.

Usage

```
## S3 method for class 'matrix'
nchains(x, ...)
```

Arguments

<code>x</code>	An object.
<code>...</code>	Other arguments passed to methods.

Value

An integer scalar of the number of chains.

See Also

Other MCMC dimensions: `niters()`, `npars()`, `nsams()`, `nsims()`, `nterms()`

nchains.marray	<i>Number of Chains</i>
----------------	-------------------------

Description

Gets the number of chains of an MCMC object.

Usage

```
## S3 method for class 'marray'
nchains(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

An integer scalar of the number of chains.

See Also

Other MCMC dimensions: [niters\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

nchains.mcmcarray	<i>Number of Chains</i>
-------------------	-------------------------

Description

Gets the number of chains of an MCMC object.

Usage

```
## S3 method for class 'mcmcarray'
nchains(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

An integer scalar of the number of chains.

See Also

Other MCMC dimensions: [niters\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

nchains.mcmc	<i>Number of Chains</i>
--------------	-------------------------

Description

Gets the number of chains of an MCMC object.

Usage

```
## S3 method for class 'mcmc'  
nchains(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

An integer scalar of the number of chains.

See Also

Other MCMC dimensions: [niters\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

nchains.mcmcrs	<i>Number of Chains</i>
----------------	-------------------------

Description

Gets the number of chains of an MCMC object.

Usage

```
## S3 method for class 'mcmcrs'  
nchains(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

An integer scalar of the number of chains.

See Also

Other MCMC dimensions: [niters\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

niters.matrix	<i>Number of Iterations</i>
---------------	-----------------------------

Description

Gets the number of iterations (in a chain) of an MCMC object.

Usage

```
## S3 method for class 'matrix'
niters(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

An integer scalar of the number of iterations.

See Also

Other MCMC dimensions: [nchains\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

niters.marray	<i>Number of Iterations</i>
---------------	-----------------------------

Description

Gets the number of iterations (in a chain) of an MCMC object.

Usage

```
## S3 method for class 'marray'
niters(x, ...)
```

Arguments

`x` An object.

`...` Other arguments passed to methods.

Value

An integer scalar of the number of iterations.

See Also

Other MCMC dimensions: [nchains\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

<code>niters.mcmcarray</code>	<i>Number of Iterations</i>
-------------------------------	-----------------------------

Description

Gets the number of iterations (in a chain) of an MCMC object.

Usage

```
## S3 method for class 'mcmcarray'
niters(x, ...)
```

Arguments

`x` An object.

`...` Other arguments passed to methods.

Value

An integer scalar of the number of iterations.

See Also

Other MCMC dimensions: [nchains\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

niters.mcmcr	<i>Number of Iterations</i>
--------------	-----------------------------

Description

Gets the number of iterations (in a chain) of an MCMC object.

Usage

```
## S3 method for class 'mcmcr'  
niters(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

An integer scalar of the number of iterations.

See Also

Other MCMC dimensions: [nchains\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

niters.mcmcrcs	<i>Number of Iterations</i>
----------------	-----------------------------

Description

Gets the number of iterations (in a chain) of an MCMC object.

Usage

```
## S3 method for class 'mcmcrcs'  
niters(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

An integer scalar of the number of iterations.

See Also

Other MCMC dimensions: [nchains\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

npars.marray

Number of Parameters

Description

Gets the number of parameters of an object.

The default methods returns the length of [pars\(\)](#) if none are NA, otherwise it returns NA.

Usage

```
## S3 method for class 'marray'
npars(x, scalar = NULL, ...)
```

Arguments

x	An object.
scalar	A logical scalar specifying whether to include all parameters (NULL), only scalars (TRUE) or all parameters except scalars (FALSE).
...	Other arguments passed to methods.

Value

An integer scalar of the number of parameters.

See Also

[pars\(\)](#)

Other MCMC dimensions: [nchains\(\)](#), [niters\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

Other parameters: [pars\(\)](#), [set_pars\(\)](#)

npars.mmcarray	<i>Number of Parameters</i>
----------------	-----------------------------

Description

Gets the number of parameters of an object.

The default methods returns the length of `pars()` if none are NA, otherwise it returns NA.

Usage

```
## S3 method for class 'mmcarray'
npars(x, scalar = NULL, ...)
```

Arguments

x	An object.
scalar	A logical scalar specifying whether to include all parameters (NULL), only scalars (TRUE) or all parameters except scalars (FALSE).
...	Other arguments passed to methods.

Value

An integer scalar of the number of parameters.

See Also

`pars()`

Other MCMC dimensions: `nchains()`, `nitters()`, `nsams()`, `nsims()`, `nterms()`

Other parameters: `pars()`, `set_pars()`

npars.mmc	<i>Number of Parameters</i>
-----------	-----------------------------

Description

Gets the number of parameters of an object.

The default methods returns the length of `pars()` if none are NA, otherwise it returns NA.

Usage

```
## S3 method for class 'mcmc'
npars(x, scalar = NULL, ...)
```

Arguments

x	An object.
scalar	A logical scalar specifying whether to include all parameters (NULL), only scalars (TRUE) or all parameters except scalars (FALSE).
...	Other arguments passed to methods.

Value

An integer scalar of the number of parameters.

See Also

[pars\(\)](#)

Other MCMC dimensions: [nchains\(\)](#), [niters\(\)](#), [nsams\(\)](#), [nsims\(\)](#), [nterms\(\)](#)

Other parameters: [pars\(\)](#), [set_pars\(\)](#)

npdims.mcmcarray

Number of Parameter Dimensions

Description

Gets the number of the dimensions of each parameter of an object.

The default methods returns the length of each element of [pdims\(\)](#) as an integer vector.

Usage

```
## S3 method for class 'mcmcarray'
npdims(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

A named integer vector of the number of dimensions of each parameter.

See Also

Other dimensions: [dims\(\)](#), [ndims\(\)](#), [pdims\(\)](#)

npdims.mcmcr	<i>Number of Parameter Dimensions</i>
--------------	---------------------------------------

Description

Gets the number of the dimensions of each parameter of an object.

The default methods returns the length of each element of `pdims()` as an integer vector.

Usage

```
## S3 method for class 'mcmcr'
npdims(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

A named integer vector of the number of dimensions of each parameter.

See Also

Other dimensions: `dims()`, `ndims()`, `pdims()`

nterms.mcmcarray	<i>Number of Terms</i>
------------------	------------------------

Description

Gets the number of terms of an MCMC object.

Usage

```
## S3 method for class 'mcmcarray'
nterms(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

A integer scalar of the number of terms.

See Also

Other MCMC dimensions: [nchains\(\)](#), [niters\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#)

nterms.mcmc	<i>Number of Terms</i>
-------------	------------------------

Description

Gets the number of terms of an MCMC object.

Usage

```
## S3 method for class 'mcmc'
nterms(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

A integer scalar of the number of terms.

See Also

Other MCMC dimensions: [nchains\(\)](#), [niters\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#)

nterms.mmc	<i>Number of Terms</i>
------------	------------------------

Description

Gets the number of terms of an MCMC object.

Usage

```
## S3 method for class 'mmc'
nterms(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

A integer scalar of the number of terms.

See Also

Other MCMC dimensions: [nchains\(\)](#), [niters\(\)](#), [npars\(\)](#), [nsams\(\)](#), [nsims\(\)](#)

<code>pars.mcmc</code>	<i>Parameter Names</i>
------------------------	------------------------

Description

Gets the parameter names.

Usage

```
## S3 method for class 'mcmc'
pars(x, scalar = NULL, terms = FALSE, ...)
```

Arguments

<code>x</code>	An object.
<code>scalar</code>	A logical scalar specifying whether to include all parameters (NULL), only scalars (TRUE) or all parameters except scalars (FALSE).
<code>terms</code>	A logical scalar specifying whether to provide the parameters for each term.
<code>...</code>	Other arguments passed to methods.

Value

A character vector of the names of the parameters.

See Also

Other parameters: [npars\(\)](#), [set_pars\(\)](#)

pars.mcmcrcs

*Parameter Names***Description**

Gets the parameter names.

Usage

```
## S3 method for class 'mcmcrcs'
pars(x, scalar = NULL, terms = FALSE, ...)
```

Arguments

x	An object.
scalar	A logical scalar specifying whether to include all parameters (NULL), only scalars (TRUE) or all parameters except scalars (FALSE).
terms	A logical scalar specifying whether to provide the parameters for each term.
...	Other arguments passed to methods.

Value

A character vector of the names of the parameters.

See Also

Other parameters: [npars\(\)](#), [set_pars\(\)](#)

pdims.mccarray

*Parameter Dimensions***Description**

Gets the dimensions of each parameter of an object.

Usage

```
## S3 method for class 'mccarray'
pdims(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

A named list of integer vectors of the dimensions of each parameter.

See Also

Other dimensions: [dims\(\)](#), [ndims\(\)](#), [npdims\(\)](#)

<code>pdims.mcmcarray</code>	<i>Parameter Dimensions</i>
------------------------------	-----------------------------

Description

Gets the dimensions of each parameter of an object.

Usage

```
## S3 method for class 'mcmcarray'  
pdims(x, ...)
```

Arguments

- `x` An object.
- `...` Other arguments passed to methods.

Value

A named list of integer vectors of the dimensions of each parameter.

See Also

Other dimensions: [dims\(\)](#), [ndims\(\)](#), [npdims\(\)](#)

<code>pdims.mcmc</code>	<i>Parameter Dimensions</i>
-------------------------	-----------------------------

Description

Gets the dimensions of each parameter of an object.

Usage

```
## S3 method for class 'mcmc'  
pdims(x, ...)
```

Arguments

`x` An object.
`...` Other arguments passed to methods.

Value

A named list of integer vectors of the dimensions of each parameter.

See Also

Other dimensions: [dims\(\)](#), [ndims\(\)](#), [npdims\(\)](#)

rhat.marray	<i>R-hat</i>
-------------	--------------

Description

Calculates an R-hat (potential scale reduction factor) value.

Usage

```
## S3 method for class 'marray'
rhat(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

`x` An object.
`by` A string indicating whether to determine by "term", "parameter" or "all".
`as_df` A flag indicating whether to return the results as a data frame versus a named list.
`na_rm` A flag specifying whether to ignore missing values.
`...` Other arguments passed to methods.

Details

By default the uncorrected, unfolded, univariate, split R-hat value.

Value

A number ≥ 1 indicating the rhat value.

References

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

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

rhat.mcmc	<i>R-hat</i>
-----------	--------------

Description

Calculates an R-hat (potential scale reduction factor) value.

Usage

```
## S3 method for class 'mcmc'
rhat(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default the uncorrected, unfolded, univariate, split R-hat value.

Value

A number ≥ 1 indicating the rhat value.

References

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

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

rhat.mcmc.list	<i>R-hat</i>
----------------	--------------

Description

Calculates an R-hat (potential scale reduction factor) value.

Usage

```
## S3 method for class 'mcmc.list'
rhat(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default the uncorrected, unfolded, univariate, split R-hat value.

Value

A number ≥ 1 indicating the rhat value.

References

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

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

rhat.mcmcarray	<i>R-hat</i>
----------------	--------------

Description

Calculates an R-hat (potential scale reduction factor) value.

Usage

```
## S3 method for class 'mcmcarray'
rhat(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default the uncorrected, unfolded, univariate, split R-hat value.

Value

A number ≥ 1 indicating the rhat value.

References

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

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

rhat.mcmc	<i>R-hat</i>
-----------	--------------

Description

Calculates an R-hat (potential scale reduction factor) value.

Usage

```
## S3 method for class 'mcmc'  
rhat(x, by = "all", as_df = FALSE, na_rm = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
...	Other arguments passed to methods.

Details

By default the uncorrected, unfolded, univariate, split R-hat value.

Value

A number ≥ 1 indicating the rhat value.

References

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

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

Examples

```
rhat(mcmc_example)  
rhat(mcmc_example, by = "parameter")  
rhat(mcmc_example, by = "term")  
rhat(mcmc_example, by = "term", as_df = TRUE)
```

rhat.mcmcrcs

*R-hat***Description**

Calculates an R-hat (potential scale reduction factor) value.

Usage

```
## S3 method for class 'mcmcrcs'
rhat(x, by = "all", as_df = FALSE, na_rm = FALSE, bound = FALSE, ...)
```

Arguments

x	An object.
by	A string indicating whether to determine by "term", "parameter" or "all".
as_df	A flag indicating whether to return the results as a data frame versus a named list.
na_rm	A flag specifying whether to ignore missing values.
bound	flag specifying whether to bind mcmcrcs objects by their chains before calculating rhat.
...	Other arguments passed to methods.

Details

By default the uncorrected, unfolded, univariate, split R-hat value.

Value

A number ≥ 1 indicating the rhat value.

References

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

See Also

Other convergence: [converged\(\)](#), [converged_pars\(\)](#), [converged_terms\(\)](#), [esr\(\)](#), [esr_pars\(\)](#), [esr_terms\(\)](#), [rhat_pars\(\)](#), [rhat_terms\(\)](#)

Examples

```
rhat(mcmcrcs(mcmcr_example, mcmcr_example))
rhat(mcmcrcs(mcmcr_example, mcmcr_example), bound = TRUE)
```

set_pars.mcmc	<i>Set Parameters</i>
---------------	-----------------------

Description

Sets an object's parameter names.

The assignment version `pars<-( )` forwards to `set_pars( )`.

Usage

```
## S3 method for class 'mcmc'
set_pars(x, value, ...)
```

Arguments

<code>x</code>	An object.
<code>value</code>	A character vector of the new parameter names.
<code>...</code>	Other arguments passed to methods.

Details

`value` must be a unique character vector of the same length as the object's parameters.

Value

The modified object.

See Also

Other parameters: [npars\(\)](#), [pars\(\)](#)

set_pars.mcmc	<i>Set Parameters</i>
---------------	-----------------------

Description

Sets an object's parameter names.

The assignment version `pars<-( )` forwards to `set_pars( )`.

Usage

```
## S3 method for class 'mcmc'
set_pars(x, value, ...)
```

Arguments

x	An object.
value	A character vector of the new parameter names.
...	Other arguments passed to methods.

Details

value must be a unique character vector of the same length as the object's parameters.

Value

The modified object.

See Also

Other parameters: [npars\(\)](#), [pars\(\)](#)

split_chains.mcmcarray

Split Chains

Description

Splits each of an MCMC object's chains in half to double the number of chains and halve the number of iterations.

Usage

```
## S3 method for class 'mcmcarray'  
split_chains(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

The modified object.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [collapse_chains\(\)](#), [estimates\(\)](#)

split_chains.mcmc	<i>Split Chains</i>
-------------------	---------------------

Description

Splits each of an MCMC object's chains in half to double the number of chains and halve the number of iterations.

Usage

```
## S3 method for class 'mcmc'
split_chains(x, ...)
```

Arguments

x	An object.
...	Other arguments passed to methods.

Value

The modified object.

See Also

Other MCMC manipulations: [bind_chains\(\)](#), [bind_iterations\(\)](#), [collapse_chains\(\)](#), [estimates\(\)](#)

subset	<i>Subset an MCMC object</i>
--------	------------------------------

Description

Subsets an MCMC object by its chains, iterations and/or parameters.

Usage

```
## S3 method for class 'mcmcarray'
subset(x, chains = NULL, iters = NULL, iterations = NULL, ...)

## S3 method for class 'mcmc'
subset(
  x,
  chains = NULL,
  iters = NULL,
  pars = NULL,
  iterations = NULL,
```

```

    parameters = NULL,
    ...
)

## S3 method for class 'mcmcrs'
subset(
  x,
  chains = NULL,
  iters = NULL,
  pars = NULL,
  iterations = NULL,
  parameters = NULL,
  ...
)

```

Arguments

x	The MCMC object to subset
chains	An integer vector of chains.
iters	An integer vector of iterations.
iterations	An integer vector (or NULL) of the iterations to subset by.
...	Unused.
pars	A character vector of parameter names.
parameters	A character vector (or NULL) of the parameters to subset by.

Methods (by class)

- `subset(mcmcarray)`: Subset an `mcmcarray` object
- `subset(mcmcr)`: Subset an `mcmcr` object
- `subset(mcmcrs)`: Subset an `mcmcrs` object

See Also

[`universals::split\_chains\(\)`](#)

Examples

```

subset(mcmcr_example,
  chains = 2L, iters = 1:100,
  pars = c("beta", "alpha")
)

```

vld_mcmcr	<i>Validate MCMC objects</i>
-----------	------------------------------

Description

Validates class and structure of MCMC objects.

Usage

```
vld_mcmarray(x)
```

```
vld_mcmcr(x)
```

```
vld_mmcrrs(x)
```

Arguments

x The object to check.

Details

To just validate class use [chk::vld_s3_class\(\)](#).

Value

A flag indicating whether the object was validated.

Functions

- [vld_mcmarray\(\)](#): Validate [mcmarray-object\(\)](#)
- [vld_mcmcr\(\)](#): Validate [mcmcr-object\(\)](#)
- [vld_mmcrrs\(\)](#): Validate [mmcrrs-object\(\)](#)

See Also

[chk_mcmcr\(\)](#)

Examples

```
#' vld_mcmarray
vld_mcmarray(1)

# vld_mcmcr
vld_mcmcr(1)
vld_mcmcr(mcmcr::mcmcr_example)

# vld_mmcrrs
vld_mmcrrs(1)
```

zero	<i>Zero MCMC sample values</i>
------	--------------------------------

Description

Zeros an MCMC object's sample values.

Usage

```
zero(x, ...)
```

```
## S3 method for class 'marray'
```

```
zero(x, ...)
```

```
## S3 method for class 'mcmarray'
```

```
zero(x, ...)
```

```
## S3 method for class 'mcmcr'
```

```
zero(x, pars = NULL, ...)
```

Arguments

x	The MCMC object.
...	Unused.
pars	A character vector (or NULL) of the pars to zero.

Details

It is used for removing the effect of a random effect where the expected value is 0.

Value

The MCMC

Methods (by class)

- `zero(marray)`: Zero an marray object
- `zero(mcmarray)`: Zero an mcmarray object
- `zero(mcmcr)`: Zero an mcmcr object

Examples

```
zero(mcmcr_example, pars = "beta")
```

Index

- * **bind**
 - bind_dimensions, 16
 - bind_dimensions_n, 16
 - bind_parameters, 17
- * **coerce**
 - as.marray, 4
 - as.mmcarray, 8
 - as.mmc, 8
 - mcrcs, 48
- * **combine**
 - combine_dimensions, 22
 - combine_samples, 23
 - combine_samples_n, 23
- * **datasets**
 - mcrcr_example, 48
- * **is**
 - is.marray, 43
 - is.mmcarray, 44
 - is.mmc, 44
 - is.mmcrcs, 45
- * **manipulate**
 - mcrc_aperm, 46
 - mcrc_map, 46
- * **objects**
 - mcrcarray-object, 47
 - mcrc-object, 47
 - mcrcs-object, 49
- as.marray, 4
- as.marray(), 8, 9, 48
- as.mmc.marray, 4
- as.mmc.mmc, 5
- as.mmc.mmcarray, 6
- as.mmc.mmc, 7
- as.mmcarray, 8
- as.mmcarray(), 4, 9, 48
- as.mmc, 8
- as.mmc(), 4, 8, 48
- as.mmc.list(as.marray), 4
- as.mmcrcs, 10
- as_nlist(), 12
- as_nlist.mmc, 11
- as_nlists(), 11
- as_nlists.mmc, 12
- bind_chains, 22, 33–35, 70, 71
- bind_chains.marray, 13
- bind_chains.mmc, 13
- bind_chains.mmc.list, 14
- bind_chains.mmcarray, 14
- bind_chains.mmc, 15
- bind_dimensions, 16
- bind_dimensions(), 17
- bind_dimensions_n, 16
- bind_dimensions_n(), 16, 17
- bind_iterations, 13–15, 22, 33–35, 70, 71
- bind_parameters, 17
- bind_parameters(), 16, 17
- check_mmcarray, 18
- check_mmc, 18
- check_mmc(), 18, 19
- chk::chk_s3_class(), 19
- chk::vld_s3_class(), 73
- chk_mmcarray(chk_mmc), 19
- chk_mmc, 19
- chk_mmc(), 73
- chk_mmcrcs(chk_mmc), 19
- coda::line(), 48
- coda::mmc(), 9
- coda::mmc.list(), 9
- coef, 20
- collapse_chains, 13–15, 33–35, 70, 71
- collapse_chains.mmc, 21
- combine_dimensions, 22
- combine_dimensions(), 23, 24
- combine_samples, 23
- combine_samples(), 22, 24
- combine_samples_n, 23
- combine_samples_n(), 22, 23

- converged, [27–31](#), [64–68](#)
- converged.default, [24](#)
- converged.mcmc, [25](#)
- converged_pars, [25–31](#), [64–68](#)
- converged_terms, [25–31](#), [64–68](#)
- dims, [57](#), [58](#), [62](#), [63](#)
- esr, [25](#), [26](#), [64–68](#)
- esr(), [32](#)
- esr.marray, [26](#)
- esr.mcmc, [27](#)
- esr.mcmc.list, [28](#)
- esr.mcmcarray, [29](#)
- esr.mcmcr, [30](#)
- esr.mmc, [31](#)
- esr_pars, [25–31](#), [64–68](#)
- esr_terms, [25–31](#), [64–68](#)
- ess, [32](#)
- estimates, [13–15](#), [22](#), [70](#), [71](#)
- estimates.marray, [32](#)
- estimates.mcmc, [33](#)
- estimates.mcmc.list, [34](#)
- estimates.mcmcarray, [34](#)
- estimates.mcmcr, [35](#)
- extras::directional_information(), [21](#)
- extras::svalue(), [21](#)
- fill_all(), [40–42](#)
- fill_all.marray, [36](#)
- fill_all.mcmcarray, [37](#)
- fill_all.mcmcr, [38](#)
- fill_na(), [36](#), [38](#), [39](#)
- fill_na.marray, [39](#)
- fill_na.mcmcarray, [41](#)
- fill_na.mcmcr, [42](#)
- is.marray, [43](#)
- is.marray(), [44](#), [45](#)
- is.mcmcarray, [44](#)
- is.mcmcarray(), [43](#), [45](#)
- is.mcmcr, [44](#)
- is.mcmcr(), [43–45](#)
- is.mmc, [45](#)
- is.mmc(), [43–45](#)
- mcmc.list, [5–8](#)
- mcmc_aperm, [46](#)
- mcmc_aperm(), [47](#)
- mcmc_map, [46](#)
- mcmc_map(), [46](#)
- mcmcarray-object, [47](#)
- mcmcarray_object (mcmcarray-object), [47](#)
- mcmcr-object, [47](#)
- mcmcr_example, [48](#)
- mcmcr_object (mcmcr-object), [47](#)
- mcmc, [48](#)
- mcmc(), [4](#), [8](#), [9](#)
- mcmc-object, [49](#)
- mcmc_object (mcmc-object), [49](#)
- mcmcUpgrade, [5–8](#)
- nchains, [52–57](#), [59](#), [60](#)
- nchains.matrix, [49](#)
- nchains.marray, [50](#)
- nchains.mcmcarray, [50](#)
- nchains.mcmcr, [51](#)
- nchains.mmc, [51](#)
- ndims, [57](#), [58](#), [62](#), [63](#)
- niters, [49–52](#), [55–57](#), [59](#), [60](#)
- niters.matrix, [52](#)
- niters.marray, [52](#)
- niters.mcmcarray, [53](#)
- niters.mcmcr, [54](#)
- niters.mmc, [54](#)
- nlist_object(), [11](#)
- nlists_object(), [12](#)
- npars, [49–55](#), [59–61](#), [69](#), [70](#)
- npars.marray, [55](#)
- npars.mcmcarray, [56](#)
- npars.mcmcr, [56](#)
- npdims, [62](#), [63](#)
- npdims.mcmcarray, [57](#)
- npdims.mcmcr, [58](#)
- nsams, [49–57](#), [59](#), [60](#)
- nsims, [49–57](#), [59](#), [60](#)
- nterms, [49–57](#)
- nterms.mcmcarray, [58](#)
- nterms.mcmcr, [59](#)
- nterms.mmc, [59](#)
- pars, [55–57](#), [69](#), [70](#)
- pars(), [55–57](#)
- pars.mcmcr, [60](#)
- pars.mmc, [61](#)
- pdims, [57](#), [58](#)
- pdims(), [57](#), [58](#)
- pdims.marray, [61](#)

pdims.mcmcarray, 62
pdims.mcmcr, 62
plot.mcmc, 5–8

rhat, 25–31
rhat.marray, 63
rhat.mcmc, 64
rhat.mcmc.list, 65
rhat.mcmcarray, 66
rhat.mcmcr, 67
rhat.mmcrcs, 68
rhat_pars, 25–31, 64–68
rhat_terms, 25–31, 64–68

set_pars, 55–57, 60, 61
set_pars.mcmcr, 69
set_pars.mmcrcs, 69
split_chains, 13–15, 22, 33–35
split_chains.mcmcarray, 70
split_chains.mcmcr, 71
stats::coef, 21
subset, 71
summary.mcmc, 5–8

thin, 5–8

universals::bind_chains(), 16, 17
universals::esr, 32
universals::split_chains(), 72

vld_mcmcarray(vld_mcmcr), 73
vld_mcmcr, 73
vld_mcmcr(), 20
vld_mmcrcs(vld_mcmcr), 73

window.mcmc, 5–8

zero, 74