

# Package ‘universals’

September 22, 2022

**Title** S3 Generics for Bayesian Analyses

**Version** 0.0.5

**Description** Provides S3 generic methods and some default implementations for Bayesian analyses that generate Markov Chain Monte Carlo (MCMC) samples. The purpose of 'universals' is to reduce package dependencies and conflicts. The 'nlist' package implements many of the methods for its 'nlist' class.

**License** MIT + file LICENSE

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

**BugReports** <https://github.com/poissonconsulting/universals/issues>

**Depends** R (>= 3.5)

**Suggests** covr, nlist, testthat

**Encoding** UTF-8

**Language** en-US

**RoxygenNote** 7.2.1

**NeedsCompilation** no

**Author** Joe Thorley [aut, cre] (<<https://orcid.org/0000-0002-7683-4592>>),  
Kirill Müller [ctb] (<<https://orcid.org/0000-0002-1416-3412>>),  
Poisson Consulting [cph, fnd]

**Maintainer** Joe Thorley <joe@poissonconsulting.ca>

**Repository** CRAN

**Date/Publication** 2022-09-22 14:10:02 UTC

## R topics documented:

bind_chains . . . . .	2
bind_iterations . . . . .	3
collapse_chains . . . . .	3
converged . . . . .	4

converged_pars . . . . .	4
converged_terms . . . . .	5
dims . . . . .	6
esr . . . . .	7
esr_pars . . . . .	8
esr_terms . . . . .	9
estimates . . . . .	10
nchains . . . . .	10
ndims . . . . .	11
niters . . . . .	12
npars . . . . .	12
npdims . . . . .	13
nsams . . . . .	14
nsims . . . . .	14
nterms . . . . .	15
pars . . . . .	16
pdims . . . . .	16
rhat . . . . .	17
rhat_pars . . . . .	17
rhat_terms . . . . .	18
set_pars . . . . .	19
split_chains . . . . .	20

**Index** **21**

---

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

---

**Description**

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

**Usage**

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_iterations	<i>Bind Iterations</i>
-----------------	------------------------

---

**Description**

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

**Usage**

```
bind_iterations(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\\_chains\(\)](#), [collapse\\_chains\(\)](#), [estimates\(\)](#), [split\\_chains\(\)](#)

---

collapse_chains	<i>Collapse Chains</i>
-----------------	------------------------

---

**Description**

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

**Usage**

```
collapse_chains(x, ...)
```

```
## Default S3 method:  
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\(\)](#)

---

converged	<i>Converged</i>
-----------	------------------

---

**Description**

Tests whether an object has converged.

**Usage**

```
converged(x, ...)
```

**Arguments**

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

**Value**

A logical scalar indicating whether the object has converged.

**See Also**

Other convergence: [converged\\_pars\(\)](#), [converged\\_terms\(\)](#), [esr\\_pars\(\)](#), [esr\\_terms\(\)](#), [esr\(\)](#), [rhat\\_pars\(\)](#), [rhat\\_terms\(\)](#), [rhat\(\)](#)

---

converged_pars	<i>Converged Parameters</i>
----------------	-----------------------------

---

**Description**

Tests whether each parameter of an object has converged.

**Usage**

```
converged_pars(x, ...)
```

**Arguments**

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

**Value**

A uniquely named logical vector indicating whether each parameter has converged.

**See Also**

Other convergence: [converged\\_terms\(\)](#), [converged\(\)](#), [esr\\_pars\(\)](#), [esr\\_terms\(\)](#), [esr\(\)](#), [rhat\\_pars\(\)](#), [rhat\\_terms\(\)](#), [rhat\(\)](#)

---

<code>converged_terms</code>	<i>Converged Terms</i>
------------------------------	------------------------

---

**Description**

Tests whether each term of an object has converged.

**Usage**

```
converged_terms(x, ...)
```

**Arguments**

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

**Value**

A list of uniquely named logical objects with whether each term has converged.

**See Also**

Other convergence: [converged\\_pars\(\)](#), [converged\(\)](#), [esr\\_pars\(\)](#), [esr\\_terms\(\)](#), [esr\(\)](#), [rhat\\_pars\(\)](#), [rhat\\_terms\(\)](#), [rhat\(\)](#)

---

`dims`*Dimensions*

---

## Description

Gets the dimensions of an object.

## Usage

```
dims(x, ...)  
  
## Default S3 method:  
dims(x, ...)  
  
## S3 method for class 'factor'  
dims(x, ...)
```

## Arguments

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

## Details

Unlike `base::dim()`, `dims` returns the length of an atomic vector.

## Value

An integer vector of the dimensions.

## See Also

[base::dim\(\)](#)  
Other dimensions: [ndims\(\)](#), [npdims\(\)](#), [pdims\(\)](#)

## Examples

```
dims(numeric(0))  
dims(1:3)  
dims(factor("a"))  
dims(matrix(1:4, nrow = 2L))  
dims(array(1:9, dim = c(3L,1L,3L)))  
dims(ToothGrowth)  
dims(Titanic)
```

---

esr *Effective Sampling Rate*

---

### Description

Calculates the effective sampling rate (esr).

### Usage

`esr(x, ...)`

### Arguments

`x` An object.  
`...` 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\\_pars\(\)](#), [converged\\_terms\(\)](#), [converged\(\)](#), [esr\\_pars\(\)](#), [esr\\_terms\(\)](#), [rhat\\_pars\(\)](#), [rhat\\_terms\(\)](#), [rhat\(\)](#)

---

 esr\_pars

*Effective Sampling Rate for Parameters*


---

### Description

Calculates the effective sampling rate (esr) for each parameter.

### Usage

```
esr_pars(x, ...)
```

### Arguments

`x`                    An object.  
`...`                    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 uniquely named numeric vector of values between 0 and 1 indicating the esr value for each parameter.

### 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\\_pars\(\)](#), [converged\\_terms\(\)](#), [converged\(\)](#), [esr\\_terms\(\)](#), [esr\(\)](#), [rhat\\_pars\(\)](#), [rhat\\_terms\(\)](#), [rhat\(\)](#)



---

`esr_terms`*Effective Sampling Rate for Terms*

---

**Description**

Calculates the effective sampling rate (esr) for each term.

**Usage**

```
esr_terms(x, ...)
```

**Arguments**

<code>x</code>	An object.
<code>...</code>	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 list of uniquely named numeric objects with values between 0 and 1 indicating the effectively sampling rate for each term.

**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\\_pars\(\)](#), [converged\\_terms\(\)](#), [converged\(\)](#), [esr\\_pars\(\)](#), [esr\(\)](#), [rhat\\_pars\(\)](#), [rhat\\_terms\(\)](#), [rhat\(\)](#)

---

estimates	<i>Estimates</i>
-----------	------------------

---

**Description**

Calculates the estimates for an MCMC object.

**Usage**

```
estimates(x, ...)
```

**Arguments**

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

**Value**

A list of uniquely named numeric objects.

**See Also**

Other MCMC manipulations: [bind\\_chains\(\)](#), [bind\\_iterations\(\)](#), [collapse\\_chains\(\)](#), [split\\_chains\(\)](#)

---

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

---

**Description**

Gets the number of chains of an MCMC object.

**Usage**

```
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\(\)](#)

---

ndims	<i>Number of Dimensions</i>
-------	-----------------------------

---

### Description

Gets the number of dimensions of an object as returned by `dims()`.

The default methods returns the length of `dims()`.

### Usage

```
ndims(x, ...)  
  
## Default S3 method:  
ndims(x, ...)  
  
## S3 method for class 'matrix'  
ndims(x, ...)  
  
## S3 method for class 'data.frame'  
ndims(x, ...)
```

### Arguments

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

### Details

For matrices `ndims()` is always 2L.

For data frames `ndims()` is always 2L.

### Value

A integer scalar of the number of dimensions.

### See Also

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

### Examples

```
ndims(character(0))  
ndims(1:3)  
ndims(matrix(1))  
ndims(data.frame())  
ndims(array(1:9, dim = c(3,1,3)))
```

niters *Number of Iterations*

---

**Description**

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

**Usage**

```
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 *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**

```
npars(x, ...)
```

```
## Default S3 method:  
npars(x, ...)
```

**Arguments**

x                    An object.  
...                  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\(\)](#)

---

npdims

*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**

```
npdims(x, ...)
```

```
## Default S3 method:
```

```
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\(\)](#)

---

nsams	<i>Number of Samples</i>
-------	--------------------------

---

**Description**

Gets the number of sample values (simulations \* terms) of an MCMC object.

The default methods returns the product of `nsims()` and `nterms()`.

**Usage**

```
nsams(x, ...)
```

```
## Default S3 method:
```

```
nsams(x, ...)
```

**Arguments**

`x`                    An object.

`...`                Other arguments passed to methods.

**Value**

An integer scalar of the number of samples.

**See Also**

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

---

nsims	<i>Number of Simulations</i>
-------	------------------------------

---

**Description**

Gets the number of simulations (iterations \* chains) of an MCMC object.

The default methods returns the product of `nchains()` and `niters()`.

**Usage**

```
nsims(x, ...)
```

```
## Default S3 method:
```

```
nsims(x, ...)
```

**Arguments**

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

**Value**

An integer scalar of the number of simulations.

**See Also**

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

---

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

---

**Description**

Gets the number of terms of an MCMC object.

**Usage**

```
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\(\)](#)

---

pars *Parameter Names*

---

**Description**

Gets the parameter names.

**Usage**

```
pars(x, ...)
```

**Arguments**

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

**Value**

A character vector of the names of the parameters.

**See Also**

Other parameters: [npars\(\)](#), [set\\_pars\(\)](#)

---

pdims *Parameter Dimensions*

---

**Description**

Gets the dimensions of each parameter of an object.

**Usage**

```
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	<i>R-hat</i>
------	--------------

---

**Description**

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

**Usage**

```
rhat(x, ...)
```

**Arguments**

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

**Details**

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

**Value**

A number  $\geq 1$  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\\_pars\(\)](#), [converged\\_terms\(\)](#), [converged\(\)](#), [esr\\_pars\(\)](#), [esr\\_terms\(\)](#), [esr\(\)](#), [rhat\\_pars\(\)](#), [rhat\\_terms\(\)](#)

---

rhat_pars	<i>R-hat Parameters</i>
-----------	-------------------------

---

**Description**

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

**Usage**

```
rhat_pars(x, ...)
```

**Arguments**

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

**Details**

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

**Value**

An uniquely named numeric atomic with values  $\geq 1$  indicating the rhat value for each parameter.

**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\\_pars\(\)](#), [converged\\_terms\(\)](#), [converged\(\)](#), [esr\\_pars\(\)](#), [esr\\_terms\(\)](#), [esr\(\)](#), [rhat\\_terms\(\)](#), [rhat\(\)](#)

---

rhat\_terms

*R-hat Terms*

---

**Description**

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

**Usage**

```
rhat_terms(x, ...)
```

**Arguments**

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

**Details**

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

**Value**

A list of uniquely named numeric objects with values  $\geq 1$  indicating the rhat value for each term.

## 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\\_pars\(\)](#), [converged\\_terms\(\)](#), [converged\(\)](#), [esr\\_pars\(\)](#), [esr\\_terms\(\)](#), [esr\(\)](#), [rhat\\_pars\(\)](#), [rhat\(\)](#)

---

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

---

## Description

Sets an object's parameter names.

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

## Usage

```
set_pars(x, value, ...)
```

```
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`*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**

```
split_chains(x, ...)
```

**Arguments**

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

**Value**

The modified object.

**See Also**

Other MCMC manipulations: [bind\\_chains\(\)](#), [bind\\_iterations\(\)](#), [collapse\\_chains\(\)](#), [estimates\(\)](#)

# Index

- \* **MCMC dimensions**
    - nchains, 10
    - niters, 12
    - npars, 12
    - nsams, 14
    - nsims, 14
    - nterms, 15
  - \* **MCMC manipulations**
    - bind\_chains, 2
    - bind\_iterations, 3
    - collapse\_chains, 3
    - estimates, 10
    - split\_chains, 20
  - \* **convergence**
    - converged, 4
    - converged\_pars, 4
    - converged\_terms, 5
    - esr, 7
    - esr\_pars, 8
    - esr\_terms, 9
    - rhat, 17
    - rhat\_pars, 17
    - rhat\_terms, 18
  - \* **dimensions**
    - dims, 6
    - ndims, 11
    - npdims, 13
    - pdims, 16
  - \* **parameters**
    - npars, 12
    - pars, 16
    - set\_pars, 19
- base::dim(), 6  
bind\_chains, 2, 3, 4, 10, 20  
bind\_iterations, 2, 3, 4, 10, 20
- collapse\_chains, 2, 3, 3, 10, 20  
converged, 4, 5, 7–9, 17–19  
converged\_pars, 4, 4, 5, 7–9, 17–19
- converged\_terms, 4, 5, 5, 7–9, 17–19
- dims, 6, 11, 13, 16  
dims(), 11
- esr, 4, 5, 7, 8, 9, 17–19  
esr\_pars, 4, 5, 7, 8, 9, 17–19  
esr\_terms, 4, 5, 7, 8, 9, 17–19  
estimates, 2–4, 10, 20
- nchains, 10, 12–15  
nchains(), 14  
ndims, 6, 11, 13, 16  
niters, 10, 12, 13–15  
niters(), 14  
npars, 10, 12, 12, 14–16, 19  
npdims, 6, 11, 13, 16  
nsams, 10, 12, 13, 14, 15  
nsims, 10, 12–14, 14, 15  
nsims(), 14  
nterms, 10, 12–15, 15  
nterms(), 14
- pars, 13, 16, 19  
pars(), 12, 13  
pars<- (set\_pars), 19  
pdims, 6, 11, 13, 16  
pdims(), 13
- rhat, 4, 5, 7–9, 17, 18, 19  
rhat\_pars, 4, 5, 7–9, 17, 17, 19  
rhat\_terms, 4, 5, 7–9, 17, 18, 18
- set\_pars, 13, 16, 19  
split\_chains, 2–4, 10, 20