Package 'term'

January 21, 2025

Title Create, Manipulate and Query Parameter Terms

Version 0.3.6

Description Creates, manipulates, queries and repairs vectors of parameter terms. Parameter terms are the labels used to reference values in vectors, matrices and arrays. They represent the names in coefficient tables and the column names in 'mcmc' and 'mcmc.list' objects.

License MIT + file LICENSE

URL https://poissonconsulting.github.io/term/, https://github.com/poissonconsulting/term

BugReports https://github.com/poissonconsulting/term/issues

Depends R (>= 4.0)

Imports chk, extras, lifecycle, purrr, rlang, universals, vctrs

Suggests covr, testthat (>= 3.0.0)

RdMacros lifecycle

Config/Needs/website poissonconsulting/poissontemplate

Config/testthat/edition 3

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2.9000

NeedsCompilation no

Author Joe Thorley [aut, cre] (<https://orcid.org/0000-0002-7683-4592>), Kirill Müller [aut] (<https://orcid.org/0000-0002-1416-3412>), Ayla Pearson [ctb] (<https://orcid.org/0000-0001-7388-1222>), Evan Amies-Galonski [ctb] (<https://orcid.org/0000-0003-1096-2089>), Poisson Consulting [cph, fnd]

Maintainer Joe Thorley <joe@poissonconsulting.ca>

Repository CRAN

Date/Publication 2025-01-21 17:40:02 UTC

Contents

as_term *Coerce to a Term Vector*

Description

Coerces an R object to a term-vector().

Usage

```
as_term(x, ...)

as.term(x, ...)

## S3 method for class 'character'
as_term(x, repair = FALSE, normalize = repair, ...)

## S3 method for class 'numeric'
as_term(x, name = "par", ...)
```

Arguments

x	The object.
...	These dots are for future extensions and must be empty.
repair	A flag specifying whether to repair terms.
normalize	A flag specifying whether to normalize terms.
name	A string specifying the name of the parameter.

Details

as.term has been **[Soft-deprecated]** for as_term.

Methods (by class)

- as_term(character): Coerce character vector to term vector
- as_term(numeric): Coerce numeric object to term vector

See Also

term-vector() and repair_terms()

Examples

```
as_term(matrix(1:4, 2))
as_term(c("parm3[10]", "parm3[2]", "parm[2,2]", "parm[1,1]"))
```

as_term_rcrd *Coerce to a Term Record*

Description

Coerces an R object to a term_rcrd.

Usage

```
as_term_rcrd(x, ...)

## S3 method for class 'character'
as_term_rcrd(x, repair = FALSE, ...)

## S3 method for class 'numeric'
as_term_rcrd(x, name = "par", ...)

## S3 method for class 'term'
as_term_rcrd(x, repair = FALSE, ...)
```

Arguments

x	The object.
...	These dots are for future extensions and must be empty.
repair	A flag specifying whether to repair terms.
name	A string specifying the name of the parameter.

Methods (by class)

- as_term_rcrd(character): Coerce character vector to term_rcrd
- as_term_rcrd(numeric): Coerce numeric vector to term_rcrd
- as_term_rcrd(term): Coerce term vector to term_rcrd

See Also

as_term() and repair_terms()

Examples

```
as_term(matrix(1:4, 2))
as_term(c("parm3[10]", "parm3[2]", "parm[2,2]", "parm[1,1]"))
```

chk_term *Check Term or Term Record*

Description

Checks if term using `vld_term()` or `vld_term_rcrd()`.

Usage

```
chk_term(x, validate = "complete", x_name = NULL)

chk_term_rcrd(x, validate = "complete", x_name = NULL)
```

Arguments

x	The object.
validate	A string specifying the level of the validation. The possible values in order of increasing strictness are 'class', 'valid', 'consistent' and 'complete'.
x_name	A string of the name of object x or NULL.

Value

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

Functions

- `chk_term_rcrd()`: Check Term Record

Examples

```
# chk_term
x <- term("x[2]", "x[1]")
chk_term(x)
x <- c("x[2]", "x[1]")
try(chk_term(x, validate = "sorted"))

# chk_term_rcrd
x <- term_rcrd("x[2]", "x[1]")
chk_term_rcrd(x)
x <- c("x[2]", "x[1]")
try(chk_term_rcrd(x, validate = "sorted"))
```

`complete_terms` *Complete Terms*

Description

Completes an object's terms.

Usage

```
complete_terms(x, ...)

## S3 method for class 'term'
complete_terms(x, ...)

## S3 method for class 'term_rcrd'
complete_terms(x, ...)
```

Arguments

x	The object.
...	These dots are for future extensions and must be empty.

Details

It must not have any invalid or missing (NA) values.

Methods (by class)

- `complete_terms(term)`: Complete Terms for a term Vector
- `complete_terms(term_rcrd)`: Complete Terms for a term_rcrd vector

See Also

`term-vector()`, `repair_terms()` and `is_incomplete_terms()`.

Examples

```
complete_terms(term("b[3]", "b[1]", "b[2]"))
complete_terms(term("z[2,2]", "z[1,1]"))
## Not run:
complete_terms(term_rcrd("b[3]", "b[1]", "b[2]"))
complete_terms(term_rcrd("z[2,2]", "z[1,1]"))

## End(Not run)
```

consistent_term *Consistent Terms*

Description

Test whether the number of dimensions of terms in the same parameter are consistent.

Usage

```
consistent_term(x)
```

Arguments

x	The object.

Value

A logical vector indicating whether the number of dimensions is consistent.

See Also

term-vector() and npdims()

Examples

```
consistent_term(term("alpha[1]", "alpha[3]", "beta[1,1]", "beta[2,1]"))
consistent_term(term("alpha[1]", NA_term_, "beta[1,1]", "beta[2]"))
```

dims.term *Dimensions*

Description

Gets the dimensions of an object.

Usage

```
## S3 method for class 'term'
dims(x, ...)
```

Arguments

x	An object.
...	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(term("beta[1,1]"))
dims(term("beta[1,1]", "beta[1,2]"))
```

`dims.term_rcrd` *Dimensions*

Description

Gets the dimensions of an object.

Usage

```
## S3 method for class 'term_rcrd'
dims(x, ...)
```

Arguments

`x`	An object.
`...`	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(term_rcrd("beta[1,1]"))
dims(term_rcrd("beta[1,1]", "beta[1,2]"))
```

`is_incomplete_terms` *Is Incomplete Terms*

Description

Tests whether a term vector has absent elements. The vector should not require repairing.

Usage

```
is_incomplete_terms(x, ...)
```

Arguments

`x`	The object.
`...`	These dots are for future extensions and must be empty.

Value

A logical scalar indicating whether the object's terms are incomplete.

See Also

`term-vector()` and `complete_terms()`

Examples

```
is_incomplete_terms(term("b[2]"))
is_incomplete_terms(term("b[2]", "b[1]"))
is_incomplete_terms(term("b[2]", "b[1]", "b[1]"))
```

`is_inconsistent_terms` *Is Inconsistent Terms*

Description

Tests whether a term vector has inconsistent elements. Returns TRUE if includes missing or invalid terms.

Usage

```
is_inconsistent_terms(x, ...)
```

Arguments

x	The object.
...	These dots are for future extensions and must be empty.

Value

A logical scalar indicating whether the object's terms are inconsistent.

See Also

term-vector() and consistent_term()

Examples

```
is_inconsistent_terms(term("b[2]"))
is_inconsistent_terms(term("b[2]", "b[1]"))
is_inconsistent_terms(term("b[2]", "b[1,1]"))
```

---

is_term *Is Term*

Description

Tests whether an R object inherits from S3 class term.

Usage

```
is_term(x)
```

Arguments

x	The object.

Details

It does not test the validity of consistency of the term elements.

Value

A flag indicating whether the test was positive.

See Also

term-vector(), vld_term(), valid_term() and consistent_term()

Examples

```
is_term(c("parameter[2]", "parameter[10]"))
is_term(term("parameter[2]", "parameter[10]"))
```

is_term_rcrd *Is Term Record*

Description

Tests whether an R object inherits from S3 class `term_rcrd`.

Usage

```
is_term_rcrd(x)
```

Arguments

`x`	The object.

Details

It does not test the validity of consistency of the term elements.

Value

A flag indicating whether the test was positive.

See Also

`valid_term()` and `consistent_term()`

Examples

```
is_term_rcrd(new_term_rcrd())
```

NA_term_ *Missing Term*

Description

A missing term element.

Usage

```
NA_term_
```

Format

An object of class `term` (inherits from `vctrs_vctr`) of length 1.

See Also

term-vector()

Examples

```
is_term(NA_term_)
is.na(NA_term_)
```

NA_term_rcrd_ *Missing Term*

Description

A missing term element of term_rcrd type.

Usage

```
NA_term_rcrd_
```

Format

An object of class term_rcrd (inherits from vctrs_rcrd, vctrs_vctr) of length 1.

See Also

term-vector()

Examples

```
is_term_rcrd(NA_term_)
is.na(NA_term_)
```

---

new_term *Construct a New Term Object*

Description

Use this function to quickly construct a term object from a character vector, without checking the input. Use term() to repair the input.

Usage

```
new_term(x = character())
```

Arguments

x	A character vector.

See Also

new_term_rcrd()

Examples

```
new_term()
new_term(c("a", "b[1]", "b[2]"))

# Terms are not checked for validity:
new_term("r[")
repair_terms(new_term("r["))
```

---

new_term_rcrd *Construct a New Term Record Object*

Description

Use this function to quickly construct a term_rcrd object.

Usage

```
new_term_rcrd(
  x = data.frame(par = character(), dim = I(list()), stringsAsFactors = FALSE)
)
```

Arguments

x	A data frame with columns par and dim.

See Also

new_term()

Examples

```
new_term_rcrd()
## Not run:
new_term_rcrd(data.frame(
  par = c("x", "x", "y"), dim = I(list(1, 2, c(2, 2))),
  stringsAsFactors = FALSE
))

## End(Not run)
```

normalize_terms *Normalize Terms*

Description

Normalizes a term vector.

Usage

```
normalize_terms(x)
```

Arguments

x	The object.

Details

If a parameter such as b is a scalar then b[1] is replaced by b but if higher indices are included such as b[2] then b is replaced by b[1].

Value

The normalized term vector.

See Also

term-vector() and repair_terms()

Examples

```
normalize_terms(new_term(c("b", "b[3]")))
normalize_terms(new_term(c("b[1]", "a[3]")))
```

npars.term *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 'term'
npars(x, scalar = NULL, ...)
```

Arguments

x	An object.
scalar	A flag specifying whether to by default return all parameters (NULL), or only scalar parameters (TRUE) or only non-scalar parameters (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()

Examples

```
npars(term("sigma", "alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]"))
```

---

npdims.term *Number of Dimensions of Each Parameter*

Description

The terms argument is **[Defunct]**.

Usage

```
## S3 method for class 'term'
npdims(x, terms = FALSE, ...)
```

Arguments

x	An object.
terms	A flag specifying whether to get the number of dimensions for each term element.
...	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()

Examples

```
npdims(term("alpha[1]", "alpha[3]", "beta[1,1]", "beta[2,1]"))
```

`nterms.default` *Number of Terms*

Description

Gets the number of terms of an object.

Usage

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

Examples

```
nterms(term("alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]"))
nterms(term("alpha[1]", "alpha[1]", "beta[1,1]", "beta[1,1]"))
```

`nterms.term` *Number of Terms of a Term*

Description

Gets the number of terms of an MCMC object.

Usage

```
## S3 method for class 'term'
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()

Examples

```
nterms(term("alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]"))
nterms(term("alpha[1]", "alpha[1]", "beta[1,1]", "beta[1,1]"))
```

---

nterms.term_rcrd *Number of Terms of a Term Record*

Description

Gets the number of terms of an MCMC object.

Usage

```
## S3 method for class 'term_rcrd'
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()

Examples

```
nterms(as_term_rcrd(term("alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]")))
nterms(as_term_rcrd(term("alpha[1]", "alpha[1]", "beta[1,1]", "beta[1,1]")))
```

pars.character *Parameter Names*

Description

Gets the parameter names.

Usage

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

Arguments

x	An object.
scalar	A flag specifying whether to by default return all parameters (NULL), or only scalar parameters (TRUE) or only non-scalar parameters (FALSE).
...	Other arguments passed to methods.

Value

A character vector of the names of the parameters.

See Also

universals::pars

Other parameters: pars.default(), pars.term(), pars.term_rcrd(), pars_terms()

Examples

```
pars(c("a", "b[1]", "a[3]"))
```

pars.default *Parameter Names*

Description

Gets the parameter names.

Usage

```
## Default S3 method:
pars(x, scalar = NULL, ...)
```

Arguments

x	An object.
scalar	A flag specifying whether to by default return all parameters (NULL), or only scalar parameters (TRUE) or only non-scalar parameters (FALSE).
...	Other arguments passed to methods.

Value

A character vector of the names of the parameters.

See Also

universals::pars

Other parameters: pars.character(), pars.term(), pars.term_rcrd(), pars_terms()

Examples

```
pars(matrix(1:4, nrow = 2))
```

---

pars.term *Parameter Names*

Description

Gets the parameter names.

Usage

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

Arguments

x	An object.
scalar	A flag specifying whether to by default return all parameters (NULL), or only scalar parameters (TRUE) or only non-scalar parameters (FALSE).
terms	A flag specifying whether to return the parameter name for each term element.
...	Other arguments passed to methods.

Value

A character vector of the names of the parameters.

See Also

universals::pars

Other parameters: pars.character(), pars.default(), pars.term_rcrd(), pars_terms()

Examples

```
term <- term(
  "alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]",
  "beta[1,2]", "beta[2,2]", "sigma", NA
)
pars(term)
pars(term, scalar = TRUE)
pars(term, scalar = FALSE)
```

`pars.term_rcrd` *Parameter Names*

Description

Gets the parameter names.

Usage

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

Arguments

`x`	An object.
`scalar`	A flag specifying whether to by default return all parameters (NULL), or only scalar parameters (TRUE) or only non-scalar parameters (FALSE).
`...`	Other arguments passed to methods.

Value

A character vector of the names of the parameters.

See Also

universals::pars

Other parameters: `pars.character()`, `pars.default()`, `pars.term()`, `pars_terms()`

Examples

```
term <- term(
  "alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]",
  "beta[1,2]", "beta[2,2]", "sigma", NA
)
pars(term)
pars(term, scalar = TRUE)
pars(term, scalar = FALSE)
```

pars_terms *Term Parameters*

Description

Gets the name of each parameter for each term.

Usage

```
pars_terms(x, scalar = NULL, ...)
```

Arguments

x	A term vector.
scalar	A flag specifying whether to by default return all parameters (NULL), or only scalar parameters (TRUE) or only non-scalar parameters (FALSE).
...	These dots are for future extensions and must be empty.

Details

The scalar argument is **[Defunct]**.

Value

A character vector of the term parameter names.

See Also

Other parameters: `pars.character()`, `pars.default()`, `pars.term()`, `pars.term_rcrd()`

Examples

```
term <- term(
  "alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]",
  "beta[1,2]", "beta[2,2]", "sigma", NA
)
pars_terms(term)
```

pdims.term *Parameter Dimensions*

Description

Gets the dimensions of each parameter of an object.

Usage

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

Arguments

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

Details

Errors if the parameter dimensions are invalid or inconsistent.

A named list of the dimensions of each parameter can be converted into the equivalent `term-vector()` using `term()`.

Value

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

See Also

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

Examples

```
pdims(term("alpha[1]", "alpha[3]", "beta[1,1]", "beta[2,1]"))
```

pdims.term_rcrd *Parameter Dimensions*

Description

Gets the dimensions of each parameter of an object.

Usage

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

Arguments

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

Details

Errors if the parameter dimensions are inconsistent.

Value

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

See Also

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

Examples

```
pdims(as_term_rcrd(term("alpha[1]", "alpha[3]", "beta[1,1]", "beta[2,1]")))
```

---

repair_terms *Repair Terms*

Description

Repairs a terms vector.

Usage

```
repair_terms(x, normalize = TRUE)
```

Arguments

x	The object.
normalize	A flag specifying whether to normalize terms.

Details

Invalid elements are replaced by missing values and spaces removed.

Value

The repaired term vector.

See Also

`term-vector()`, `valid_term()` and `normalize_terms()`

Examples

```
repair_terms(new_term(c("b[3]", "b")))
repair_terms(new_term(c("a[3]", "b[1]")))
repair_terms(new_term(c("a [3]", " b [ 1  ] ")))
repair_terms(new_term(c("a", NA)))
```

`scalar_term` *Scalar Term*

Description

Test whether each term is a scalar.

Usage

```
scalar_term(x)
```

Arguments

`x`	The object.

Value

A logical vector indicating whether the term is a scalar.

Examples

```
scalar_term(term("alpha[1]", "alpha[3]", "beta[1]", "sigma[3]"))
scalar_term(term("alpha[1]", NA_term_, "beta[1]", "beta[3]"))
```

`set_pars.term` *Set Parameter Names*

Description

Sets an object's parameter names.

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

Usage

```
## S3 method for class 'term'
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()

Examples

```
term <- as_term(c("b[2]", "a[1]", "b[3,3]"))
set_pars(term, c("x", "y"))
```

---

subset.term *Subset Term Vector*

Description

Subsets a term vector.

Usage

```
## S3 method for class 'term'
subset(x, pars = NULL, select = NULL, ...)
```

Arguments

x	The object.
pars	A character vector of parameter names.
select	A character vector of the names of the parameters to include in the subsetted object.
...	These dots are for future extensions and must be empty.

Details

The select argument is **[Defunct]**.

Value

The modified term vector.

See Also

term-vector()

Examples

```
term <- term(
  "alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]",
  "beta[1,2]", "beta[2,2]", "sigma"
)
subset(term, "beta")
subset(term, c("alpha", "sigma"))
```

---

subset.term_rcrd *Subset Term Record*

Description

Subsets a term_rcrd.

Usage

```
## S3 method for class 'term_rcrd'
subset(x, pars = NULL, ...)
```

Arguments

x	The object.
pars	A character vector of parameter names.
...	These dots are for future extensions and must be empty.

Value

The modified term vector.

See Also

term_rcrd_object()

Examples

```
term_rcrd <- term_rcrd(
  "alpha[1]", "alpha[2]", "beta[1,1]", "beta[2,1]",
  "beta[1,2]", "beta[2,2]", "sigma"
)
## Not run:
subset(term_rcrd, "beta")
subset(term_rcrd, c("alpha", "sigma"))

## End(Not run)
```

---

term *Create Term Vector*

Description

Creates a term vector from values. A `term` vector is an S3 vector of parameter terms of the form `p`, `q[#]` or `r[#,#]` where `#` are positive integers. This function checks that all terms are valid but does not require stronger levels of consistency, see `chk_valid()` for details.

Usage

```
term(...)
```

Arguments

`...`	Unnamed values are term values, named values describe the parameter in the name and the dimensionality in the value.

Value

A term vector.

See Also

`dims()`, `ndims()`, `npdims()` and `pdims()`

Other term: `term_rcrd()`, `tindex()`

Examples

```
term()
term("p", "q[1]", "q[2]", "q[3]")
term("q[1]", "q[2]", "q[3]")
combined <- term(par = 2:4, "alpha")
pdims(combined)
term(!!!pdims(combined))

# Invalid terms are rejected:
try(term("r["))
```

```
# Valid terms are repaired
term("r  [ 1  ,2  ]")
```

`term_rcrd` *Create Term Record*

Description

Creates a term_rcrd from values. This function checks that all terms are valid but does not require stronger levels of consistency, see `chk_valid()` for details.

Usage

```
term_rcrd(...)
```

Arguments

`...`	Unnamed values are term values, named values describe the parameter in the name and the dimensionality in the value.

Value

A term_rcrd vector.

See Also

`dims()`, `ndims()`, `npdims()` and `pdims()`

Other term: `term()`, `tindex()`

Examples

```
term_rcrd()
## Not run:
term_rcrd("p", "q[1]", "q[2]", "q[3]")
term_rcrd("q[1]", "q[2]", "q[3]")

## End(Not run)
```

tindex *Term Index*

Description

Gets the index for each term of an term or term_rcrd object.

Usage

```
tindex(x)
```

Arguments

x	The object.

Details

For example the index of `beta[2,1]` is `c(2L, 1L)` while the index for `sigma` is `1L`. It is useful for extracting the values of individual terms.

Value

A named list of integer vectors of the index for each term.

See Also

`dims()`, `ndims()`, `npdims()` and `pdims()`

Other term: `term()`, `term_rcrd()`

Examples

```
tindex(term("alpha", "alpha[2]", "beta[1,1]", "beta[2 ,1  ]"))
```

valid_term *Test Valid Terms*

Description

Test whether each element in a term or term_rcrd object is valid.

Usage

```
valid_term(x)
```

Arguments

x	The object.

Details

Repairing a term vector replaces invalid terms with missing values.

Value

A logical vector indicating whether each term is valid.

See Also

`term-vector()` and `repair_terms()`

Other valid: `vld_term()`

Examples

```
# valid term elements
valid_term(term("a", "a [3]", " b [ 1  ] ", "c[1,300,10]"))
# invalid term elements
valid_term(new_term(c("a b", "a[1]b", "a[0]", "b[1,]", "c[]", "d[1][2]")))
```

---

vld_term *Validate Term or Term Record*

Description

Validates the elements of a term or term_rcrd vector.

Usage

```
vld_term(x, validate = "complete")

vld_term_rcrd(x, validate = "complete")
```

Arguments

`x`	The object.
`validate`	A string specifying the level of the validation. The possible values in order of increasing strictness are 'class', 'valid', 'consistent' and 'complete'.

Details

Internal validity of a term can be checked on three levels:

- `"valid"` checks that all terms are of the form `x`, `x[#]`, `x[#,#]` etc. where `x` is an identifier and `#` are positive integers.
- `"consistent"` checks that all terms are addressed with the same dimensionality; the terms `x[1]` and `x[2,3]` are inconsistent.

- `"complete"` checks that the values span all possible values across all dimensions; if `x[3,4]` exist, the vector must contain at least 11 more terms to be consistent (`x[1,1]` to `x[1,4]`, `x[2,1]` to `x[2,4]` and `x[3,1]` to `x[3,3]`).

Missing values are ignored as are duplicates and order.

Value

A flag indicating whether the condition was met.

Functions

- `vld_term_rcrd()`: Validate Term Record

See Also

`chk_term()`

Other valid: `valid_term()`

Other valid: `valid_term()`

Examples

```
# vld_term
vld_term(c("x[2]", "x[1]"))
vld_term(term("x[2]", "x[1]"))

# vld_term_rcrd
vld_term_rcrd(c("x[2]", "x[1]"))
vld_term_rcrd(term_rcrd("x[2]", "x[1]"))
```

Index