Package ‘narray’

October 13, 2022

Title Subset- And Name-Aware Array Utility Functions
Version 0.5.1
Author Michael Schubert <mschu.dev@gmail.com>
Maintainer Michael Schubert <mschu.dev@gmail.com>
Description Stacking arrays according to dimension names, subset-aware splitting and mapping of functions, intersecting along arbitrary dimensions, converting to and from data.frames, and many other helper functions.

URL https://github.com/mschubert/narray

BugReports https://github.com/mschubert/narray/issues

Depends R (>= 3.0.2)

LinkingTo Rcpp

SystemRequirements C++11

Imports progress, Rcpp, stats, stringr, utils

License Apache License (== 2.0) | file LICENSE

Encoding UTF-8

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

RoxygenNote 7.2.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-10-02 13:40:02 UTC

R topics documented:

bind ................................................................. 2
collect ............................................................ 3
construct ......................................................... 3
dim ................................................................. 4
bind

Binds arrays together disregarding names

Description

Binds arrays together disregarding names

Usage

\[
\text{bind}(\ldots, \text{along} = \text{length}(\text{dim(arrayList[[1]])}) + 1)
\]

Arguments

\[
\begin{align*}
\ldots & \quad \text{N-dimensional arrays, or a list thereof} \\
\text{along} & \quad \text{Along which axis to bind them together (default: new axis)}
\end{align*}
\]

Value

A joined array
**collect**

Converts a logical matrix to a list of character vectors

**Description**

This currently only supports \( x \) with only one non-zero element

**Usage**

\[
\text{collect}(x, \text{along} = 2)
\]

**Arguments**

- \( x \): A logical matrix
- \( \text{along} \): Which axis to spread mask on

**Value**

A character vector or list thereof

---

**construct**

Transform a data.frame with axes and value into an array

**Description**

The `construct()` function can be called either with the data.frame as the first argument or the formula and then specify `data=<data.frame>`

**Usage**

\[
\text{construct(data, formula = guess_structure(data), fill = NA,}
\text{ name_axes = TRUE)}
\]

**Arguments**

- \( \text{data} \): A data frame
- \( \text{formula} \): A formula: value ~ axis1 [ + axis2 + axis n ..]
- \( \text{fill} \): Value to fill array with if undefined
- \( \text{name_axes} \): Keep column names of ‘data’ as axis names

**Value**

A structured array
### dim

*base::dim, but returning 1 for vector*

**Description**

base::dim, but returning 1 for vector

**Usage**

```r
dim(x)
```

**Arguments**

- `x` Object to get dimensions on

### dimnames

*Return dimension names of an array respecting the number of dimensions*

**Description**

Act on each element if 'x' is a list

**Usage**

```r
dimnames(x, along = TRUE, null_as_integer = FALSE, drop = !identical(along, TRUE))
```

**Arguments**

- `x` An n-dimensional array
- `along` Limit to dimension (default: all)
- `null_as_integer` Whether nameless dimensions should be NULL or numbered
- `drop` Drop list of only one axis requested (default: if not returning all dimensions)

**Value**

A list of dimension names with length `length(ndim(X))`
**drop_if**

*Drop unused dims if flag is TRUE*

**Description**
Drop unused dims if flag is TRUE

**Usage**
drop_if(x, flag)

**Arguments**
- **x**: An array object
- **flag**: Whether to drop unused dimensions

**Value**
The object in full or with dropped dimensions

---

**filter**

*Function to discard subsets of an array (NA or drop)*

**Description**
Function to discard subsets of an array (NA or drop)

**Usage**
filter(X, along, FUN, subsets = base::rep(1, dim(X)[along]),
na.rm = FALSE)

**Arguments**
- **X**: An n-dimensional array
- **along**: Along which axis to apply FUN
- **FUN**: Function to apply, needs to return TRUE (keep) or FALSE
- **subsets**: Subsets that should be used when applying FUN
- **na.rm**: Whether to omit columns and rows with NAs

**Value**
An array where filtered values are NA or dropped
flatten

**Description**

Flattens an array along an axis

**Usage**

`flatten(x, along = -1, name_sep = NA)`

**Arguments**

- `x`: Array
- `along`: Along which axis to bind them together (default: last)
- `name_sep`: Which character to use for naming new arrays [default: NA, do not touch names]

**Value**

An array with n-1 dimensions

---

guess_structure

**Description**

Infer array structure from data.frame

**Usage**

`guess_structure(df, verbose = TRUE)`

**Arguments**

- `df`: A data.frame with ordered axes, value field last
- `verbose`: Print message with inferred structure (default: TRUE)

**Value**

A formula describing this structure
**intersect**

Intersects all passed arrays along a given dimension, and modifies them in place.

**Description**

TODO: accept along=c(1,2,1,1...) [maybe list w/ vectors as well?] TODO: accept data=env/list arg? [sig-comb/drug-tissue/assocs.r#62-65]

**Usage**

```r
intersect(..., along = 1, envir = parent.frame(), drop = FALSE, fail_if_empty = TRUE)
```

**Arguments**

- `...`: Arrays that should be intersected
- `along`: The axis along which to intersect
- `envir`: A list or environment to act upon
- `drop`: Drop unused dimensions on result
- `fail_if_empty`: Stop if intersection yields empty set

**intersect_list**

Intersects a list of arrays for common dimension names

**Description**

Intersects a list of arrays for common dimension names

**Usage**

```r
intersect_list(l., along = 1, drop = FALSE, fail_if_empty = TRUE)
```

**Arguments**

- `l.`: List of arrays to perform operations on
- `along`: The axis along which to intersect
- `drop`: Drop unused dimensions on result
- `fail_if_empty`: Stop if intersection yields empty set
**lambda**  
*Lambda syntax for array iteration*

**Description**

Lambda syntax for array iteration

**Usage**

```r
lambda(fml, along, group = c(), simplify = TRUE, expand_grid = TRUE, 
envir = parent.frame())
```

**Arguments**

- `fml`: A call prefixed with a tilde
- `along`: A named vector which objects to subset (eg: c(x=1))
- `group`: Not implemented
- `simplify`: Return array instead of index+result if scalar
- `expand_grid`: Use all combinations of indices (default: TRUE)
- `envir`: Environment where variables can be found

**like**  
*Reshapes x to be like like, including dimension names*

**Description**

Reshapes x to be like like, including dimension names

**Usage**

```r
like(x, like)
```

**Arguments**

- `x`: An n-dimensional array
- `like`: An n-dimensional array whose form X should inherit

**Value**

An array with values of X and structure of like
map

Maps a function along an array preserving its structure

Description
Maps a function along an array preserving its structure

Usage
map(X, along, FUN, subsets = base::rep(1, dim(X)[along]), drop = TRUE, ...)

Arguments
- X: An n-dimensional array
- along: Along which axis to apply the function
- FUN: A function that maps a vector to the same length or a scalar
- subsets: Whether to apply FUN along the whole axis or subsets thereof
- drop: Remove unused dimensions after mapping; default: TRUE
- ...: Other arguments passed to FUN

Value
An array where FUN has been applied

map_one

Apply function that preserves order of dimensions

Description
Apply function that preserves order of dimensions

Usage
map_one(X, along, FUN, pb, drop = TRUE, ...)

Arguments
- X: An n-dimensional array
- along: Along which axis to apply the function
- FUN: A function that maps a vector to the same length or a scalar
- pb: Progress bar object
- drop: Remove unused dimensions after mapping; default: TRUE
- ...: Arguments passed to the function
Value
An array where FUN has been applied

mask
Converts a list of character vectors to a logical matrix

Description
Converts a list of character vectors to a logical matrix

Usage
mask(x, along = 2, na_rm = FALSE)

Arguments
x A list of character vectors
along Which axis to spread mask on
na_rm Remove values that were translated to NAs

Value
A logical occurrence matrix

match
match() function with extended functionality

Description
match() function with extended functionality

Usage
match(x, from, to, filter_from = NULL, filter_to = NULL,
data = parent.frame(), fuzzy_level = 0, table = FALSE,
na_rm = FALSE, warn = !table && fuzzy_level > 0)
melt

Arguments

- **x**: Vector of identifiers that should be mapped
- **from**: Vector of identifiers that can be mapped
- **to**: Matched mapping for all identifiers
- **filter_from**: Restrict matching to a subset from ‘from’
- **filter_to**: Restrict matching to a subset from ‘to’
- **data**: List containing the data ‘from’ and ‘to’ reference
- **fuzzy_level**: 0 for exact, 1 punctuation, and 2 closest character
- **table**: Return a matching table instead of just the matches
- **na_rm**: Flag to remove items that can not be mapped
- **warn**: Display warning for all fuzzy matches

Value

Mapped values

---

**melt**  
*Function to melt data.frame from one or multiple arrays*

Description

Function to melt data.frame from one or multiple arrays

Usage

```r
melt(..., dimnames = NULL, na_rm = TRUE)
```

Arguments

- **...**: Array[s] or data.frame[s] to be melted
- **dimnames**: List of names along the dimensions
- **na_rm**: Remove rows with NAs

Value

`data.frame` with 'value' (or object names if multiple) indexed by axes
### named_dots

*Return a list of named dot-arguments*

**Description**

Return a list of named dot-arguments

**Usage**

```
named_dots(...)  
```

**Arguments**

...  

Function arguments

**Value**

Named function arguments

---

### narray

*R package for subset- and name-aware array utility functions*

**Description**

Stacking arrays according to dimension names, subset-aware splitting and mapping of functions, intersecting along arbitrary dimensions, converting to and from data.frames, and many other helper functions.

---

### pb

*Progress bar format to be consistent*

**Description**

Progress bar format to be consistent

**Usage**

```
pb(ticks)  
```

**Arguments**

ticks  

Number of ticks the bar has

**Value**

A progress bar object
rep

Repeats an array along an arbitrary axis

Description
Repeats an array along an arbitrary axis

Usage
rep(x, n, along = 1)
crep(x, n)
rrep(x, n)

Arguments
x
An array object
n
Integer, how often to repeat
along
Along which axis to repeat (default: 1)

Value
An array that is repeated ‘n’ times on axis ‘along’

restore_null_dimnames

If no dimnames, return NULL and not list of NULLs

Description
If no dimnames, return NULL and not list of NULLs

Usage
restore_null_dimnames(x)

Arguments
x
An array object

Value
The object with NULL if no dimnames
split

Splits and array along a given axis, either totally or only subsets

Description

Splits and array along a given axis, either totally or only subsets

Usage

split(X, along, subsets = c(1:dim(X)[along]), drop = NULL)

Arguments

- **X**: An array that should be split
- **along**: Along which axis to split; use -1 for highest dimension
- **subsets**: Whether to split each element or keep some together
- **drop**: Remove unused dimensions after mapping default: drop if all resulting arrays have same number of dimensions

Value

A list of arrays that combined make up the input array

stack

Stacks arrays while respecting names in each dimension

Description

Stacks arrays while respecting names in each dimension

Usage

stack(..., along = length(dim(arrayList[[1]])) + 1, fill = NA,
      drop = FALSE, keep_empty = FALSE, allow_overwrite = FALSE,
      fail_if_empty = TRUE)

Arguments

- **...**: N-dimensional arrays, or a list thereof
- **along**: Which axis arrays should be stacked on (default: new axis)
- **fill**: Value for unknown values (default: NA)
- **drop**: Drop unused dimensions (default: FALSE)
- **keep_empty**: Keep empty elements when stacking (default: FALSE)
- **allow_overwrite**: Overwrite values if more arrays share same key
- **fail_if_empty**: Stop if no arrays left after removing empty elements
**subset**

A stacked array, either n or n+1 dimensional

---

**Description**

Subsets an array using a list with indices or names

**Usage**

`subset(X, index, along = -1, drop = FALSE)`

**Arguments**

- `X` The array to subset
- `index` A list of vectors to use for subsetting, or vector if along is given
- `along` Along which dimension to subset if index is a vector; default is last dimension; argument is ignored if X is a vector
- `drop` Remove unused dimensions after mapping; default: TRUE

**Value**

The subset of the array

---

**translate**

Translate an axis between two sets of identifiers

---

**Description**

Translate an axis between two sets of identifiers

**Usage**

`translate(x, along = 1, to, from = dimnames(x)[[along]], ..., FUN, na.rm = FALSE)`

**Arguments**

- `x` A matrix
- `along` Along which axis to summarize
- `to` Names that this dimension should be summarized to
- `from` Names that match the dimension ‘along’
- `...` Parameters passed to ‘match’
- `FUN` Which function to apply, default is throwing error on aggregation
- `na.rm` Remove values that were translated to NAs
Value

A summarized matrix as defined by ‘from’, ‘to’

vectors_to_row_or_col  Converts vectors in a list to row- or column vectors

Description

Converts vectors in a list to row- or column vectors

Usage

vectors_to_row_or_col(xlist, along)

Arguments

xlist  List of array-like elements and vectors
along  Along which dimension vectors should be aligned

Value

List where vectors are replaced by row- or col vectors (2d)

which  A multidimensional which function

Description

A multidimensional which function

Usage

which(x, drop = TRUE)

Arguments

x  N-dimensional logical array
drop  Return a vector if called on a vector

Value

A matrix with indices where A == TRUE
Operator for array-like logical operations

Description
Operator for array-like logical operations

Usage
a %or% b

Arguments
a  First vector
b  Second vector

Value
TRUE/FALSE for each element
Index

%or%, 17  
bind, 2  
collect, 3  
construct, 3  
crep (rep), 13  
dim, 4  
dimnames, 4  
drop_if, 5  
filter, 5  
flatten, 6  
guess_structure, 6  
intersect, 7  
intersect_list, 7  
lambda, 8  
like, 8  
map, 9  
map_one, 9  
mask, 10  
match, 10  
melt, 11  
named_dots, 12  
narray, 12  
narray-package (narray), 12  
pb, 12  
rep, 13  
restore_null_dimnames, 13  
rrep (rep), 13  
split, 14  
stack, 14  
subset, 15  

translate, 15  
vectors_to_row_or_col, 16  
which, 16