Package 'multiApply'

November 20, 2018

Title Apply Functions to Multiple Multidimensional Arrays or Vectors

Version 2.0.1

Description The base apply function and its variants, as well as the related functions in the 'plyr' package, typically apply user-defined functions to a single argument (or a list of vectorized arguments in the case of mapply). The 'multiApply' package extends this paradigm with its only function, Apply, which efficiently applies functions taking one or a list of multiple unidimensional or multidimensional numeric arrays (or combinations thereof) as input. The input arrays can have different numbers of dimensions as well as different dimension lengths, and the applied function can return one or a list of unidimensional or multidimensional arrays as output. This saves development time by preventing the R user from writing often error-prone and memory-inefficient loops dealing with multiple complex arrays. Also, a remarkable feature of Apply is the transparent use of multi-core through its parameter 'ncores'. In contrast to the base apply function, this package suggests the use of 'target dimensions' as opposite to the 'margins' for specifying the dimensions relevant to the function to be applied.

Depends R (>= 3.2.0)

Imports doParallel, foreach, plyr

Suggests testthat

License LGPL-3

URL https://earth.bsc.es/gitlab/ces/multiApply

BugReports https://earth.bsc.es/gitlab/ces/multiApply/issues

Encoding UTF-8

LazyData true

RoxygenNote 5.0.0

NeedsCompilation no

Author BSC-CNS [aut, cph],
Nicolau Manubens [aut],
Alasdair Hunter [aut],
Nuria Perez [cre]

Maintainer Nuria Perez <nuria.perez@bsc.es>

R topics documented:

`Apply` *Apply Functions to Multiple Multidimensional Arrays or Vectors*

Description

This function efficiently applies a given function, which takes N vectors or multi-dimensional arrays as inputs (which may have different numbers of dimensions and dimension lengths), and applies it to a list of N vectors or multi-dimensional arrays with at least as many dimensions as expected by the given function. The user can specify which dimensions of each array the function is to be applied over with the `margins` or `target_dims` parameters. The function to be applied can receive other helper parameters and return any number of numeric vectors or multidimensional arrays. The target dimensions or margins can be specified by their names, as long as the inputs are provided with dimension names (recommended). This function can also use multi-core in a transparent way if requested via the `ncores` parameter.

The following steps help to understand how `Apply` works:

- The function receives N arrays with Dn dimensions each.
- The user specifies, for each of the arrays, which of its dimensions are 'target' dimensions (dimensions which the function provided in 'fun' operates with) and which are 'margins' (dimensions to be looped over).
- `Apply` will generate an array with as many dimensions as margins in all of the input arrays. If a margin is repeated across different inputs, it will appear only once in the resulting array.
- For each element of this resulting array, the function provided in the parameter'fun' is applied to the corresponding sub-arrays in 'data'.
- If the function returns a vector or a multidimensional array, the additional dimensions will be prepended to the resulting array (in left-most positions).
- If the provided function returns more than one vector or array, the process above is carried out for each of the outputs, resulting in a list with multiple arrays, each with the combination of all target dimensions (at the right-most positions) and resulting dimensions (at the left-most positions).

Usage

```
Apply(data, target_dims = NULL, fun, ..., output_dims = NULL,
  margins = NULL, guess_dim_names = TRUE, ncores = NULL,
  split_factor = 1)
```

Arguments

`data`	One or a list of numeric object (vector, matrix or array). They must be in the same order as expected by the function provided in the parameter 'fun'. The dimensions do not necessarily have to be ordered. If the 'target_dims' require a different order than the provided, `Apply` will automatically reorder the dimensions as needed.
`target_dims`	One or a list of vectors (or NULLs) containing the dimensions to be input into fun for each of the objects in the data. If a single vector of target dimensions is specified and multiple inputs are provided in 'data, then the single set of target dimensions is re-used for all of the inputs. These vectors can contain either integers specifying the position of the dimensions, or character strings corresponding to the dimension names. This parameter is mandatory if 'margins' are

not specified. If both 'margins' and 'target_dims' are specified, 'margins' takes priority.

`fun` Function to be applied to the arrays. Must receive as many inputs as provided in 'data', each with as many dimensions as specified in 'target_dims' or as the total number of dimensions in 'data' minus the ones specified in 'margins'. The function can receive other additional fixed parameters (see parameter '...' of `Apply`). The function can return one or a list of numeric vectors or multidimensional arrays, optionally with dimension names which will be propagated to the final result. The returned list can optionally be named, with a name for each output, which will be propagated to the resulting array. The function can optionally be provided with the attributes 'target_dims' and 'output_dims'. In that case, the corresponding parameters of `Apply` do not need to be provided. The function can expect named dimensions for each of its inputs, in the same order as specified in 'target_dims' or, if no 'target_dims' have been provided, in the same order as provided in 'data'.

`...` Additional fixed arguments expected by the function provided in the parameter 'fun'.

`output_dims` Optional list of vectors containing the names of the dimensions to be output from the fun for each of the objects it returns (or a single vector if the function has only one output).

`margins` One or a list of vectors (or NULLs) containing the 'margin' dimensions to be looped over for each input in 'data'. If a single vector of margins is specified and multiple inputs are provided in 'data', then the single set of margins is re-used for all of the inputs. These vectors can contain either integers specifying the position of the margins, or character strings corresponding to the dimension names. If both 'margins' and 'target_dims' are specified, 'margins' takes priority.

`guess_dim_names` Whether to automatically guess missing dimension names for dimensions of equal length across different inputs in 'data' with a warning (TRUE; default), or to crash whenever unnamed dimensions of equa length are identified across different inputs (FALSE).

`ncores` The number of parallel processes to spawn for the use for parallel computation in multiple cores.

`split_factor` Factor telling to which degree the input data should be split into smaller pieces to be processed by the available cores. By default (split_factor = 1) the data is split into 4 pieces for each of the cores (as specified in ncores). A split_factor of 2 will result in 8 pieces for each of the cores, and so on. The special value 'greatest' will split the input data into as many pieces as possible.

Details

When using a single object as input, Apply is almost identical to the apply function (as fast or slightly slower in some cases; with equal or improved -smaller- memory footprint).

Value

List of arrays or matrices or vectors resulting from applying 'fun' to 'data'.

References

Wickham, H (2011), The Split-Apply-Combine Strategy for Data Analysis, Journal of Statistical Software.

Examples

```
#Change in the rate of exceedance for two arrays, with different
#dimensions, for some matrix of exceedances.
data <- list(array(rnorm(1000), c(5, 10, 20)),
             array(rnorm(500), c(5, 10, 10)),
             array(rnorm(50), c(5, 10)))
test_fun <- function(x, y, z) {
  ((sum(x > z) / (length(x))) /
  (sum(y > z) / (length(y)))) * 100
}
test <- Apply(data, target = list(3, 3, NULL), test_fun)
```

Index