

# The Oarray Package

May 9, 2008

**Version** 1.4-2

**Date** 2007-10-03

**Title** Arrays with arbitrary offsets

**Author** Jonathan Rougier <j.c.rougier@bristol.ac.uk>

**Description** Generalise the starting point of the array index

**License** GPL

**Maintainer** Jonathan Rougier <j.c.rougier@bristol.ac.uk>

## R topics documented:

Oarray . . . . .	1
internal . . . . .	3
<b>Index</b>	<b>5</b>

---

Oarray                      *Arrays with arbitrary offsets*

---

## Description

The traditional R array has extents which are indexed with integers that start at 1. This is generalized to arbitrary offsets, where extent `i` is indexed with integers that start at `offset[i]`, which must be no less than zero to accomodate the R convention of dropping components with negative indices. In order to use negative offsets, the flag `drop.negative` can be set `FALSE`.

**Usage**

```
Oarray(data=NA, dim=length(data), dimnames=NULL, offset=rep(1, length(dim)),
       drop.negative=TRUE)
as.Oarray(x, offset=rep(1, length(dim)), drop.negative=TRUE)
## S3 method for class 'Oarray':
as.array(x, ...)
## S3 method for class 'Oarray':
print(x, ...)
```

**Arguments**

<code>data</code> , <code>dim</code> , <code>dimnames</code> , <code>drop</code>	As in the function <code>array</code>
<code>offset</code>	Vector of first index values for each extent (defaults to 1s); a length-one argument will be silently recycled to the appropriate length
<code>drop.negative</code>	Should negative subscripts indicate exclusion?
<code>x</code>	An array, possibly of class ‘Oarray’
<code>...</code>	Additional arguments to <code>print</code> or <code>as.array()</code>

**Details**

Typical uses are

```
x[i, j]
x[i, j] <- someValues
```

where `x` is an object of class ‘Oarray’ and `i`, `j` are indices specifying which elements to extract or replace.

Indexing may be via a logical matrix, which indicates which elements to select.

Indexing may be via a single numeric matrix with the one column for each dimension: the offset is sweep()-ed out. See `Extract.Rd` for details.

The use of `drop.negative = FALSE` will only work in `[.Oarray]` where it is provided as the final argument inside the square brackets.

**Value**

Typically an array with or without a ‘Oarray’ class attribute. Extracting from an ‘Oarray’ object unclasses the result which is then a simple array, but assigning into an ‘Oarray’ object leaves the result as an ‘Oarray’ object.

The print method provides more informative extent labelling in the case where `dimnames` are not provided.

**Side effects**

The function `base::as.array` is redefined as generic, to provide an `as.array.Oarray` method.

**Author(s)**

Jonathan Rougier, (j.c.rougier@bristol.ac.uk)

**See Also**

[array](#), [Extract](#)

**Examples**

```
fred <- Oarray(1:24, 2:4, list(c("sad", "happy"), NULL, NULL),
  offset=rep(7, 3))

tmp <- as.array(fred)
fred1 <- as.Oarray(tmp, offset=rep(7, 3))
stopifnot(identical(fred, fred1))

print.default(fred) # print method provides numbers for
fred              # non-named extents

# examples of extraction

fred[] # unclasses fred
fred["sad", 7, -9]
fred["sad", 7, -9, drop=FALSE]
fred[-8, , 7:8]

i <- 8:9; fred[, , i+1]
how.I.feel <- "happy"; fred[how.I.feel, , -(7:8)]

# examples of assignment

fred["sad", 7, -9] <- NA
fred[, , i] <- 100
fred[how.I.feel, , -(7:8)] <- Inf

# now use negative offsets and suppress usual behaviour

fred <- Oarray(24:1, 2:4, offset=c(-1, -2, 7), drop.negative=FALSE)
fred[] <- 1:24
fred[-(1:0), , 7:8]
fred[-(1:0), , 7:8] <- 100
dimnames(fred) <- list(c("sad", "happy"), NULL, NULL)
fred["sad", -2, 10] <- NA

# array and logical indexing

a <- Oarray(0, dim=rep(2,4), offset=rep(0,4))
a[diag(4)] <- 1

a[a == 0] <- NA
```

---

internal

*Internal functions for package Oarray*

---

## Description

Internal functions for package Oarray.

## Usage

```
.handleTheOffset(mc, dim, offset, dn)
```

## Arguments

mc	Index set
dim	Dimensions of object
offset	Offset for index set
dn	Logical: drop if negative?

## Details

For versions <2.8.0, the base function `as.array` is redefined as generic to allow it to have method `Oarray`.

`.handleTheOffset` is the workhorse function that is called by the subsetting functions `[.Oarray` and `[<-.Oarray`.

## Value

`.handleTheOffset` returns an object with indices in the original values (ie with offset one) to be evaluated.

`as.array.default` is simply `as.array` from the base.

## Author(s)

Jonathan Rougier, (j.c.rougier@bristol.ac.uk)

## Examples

```
#### nothing to run here
```

# Index

\*Topic **array**

`Oarray`, 1

\*Topic **internal**

`internal`, 3

`.handleTheOffset (internal)`, 3

`[.Oarray (Oarray)`, 1

`[<-.Oarray (Oarray)`, 1

`array`, 2

`as.array (internal)`, 3

`as.array.Oarray (Oarray)`, 1

`as.Oarray (Oarray)`, 1

`Extract`, 2

`internal`, 3

`is.Oarray (Oarray)`, 1

`Oarray`, 1

`print.Oarray (Oarray)`, 1