Package ‘magclass’

October 13, 2022

Type Package

Title Data Class and Tools for Handling Spatial-Temporal Data

Version 6.0.9

Date 2021-09-29

Description Data class for increased interoperability working with spatial-temporal data together with corresponding functions and methods (conversions, basic calculations and basic data manipulation). The class distinguishes between spatial, temporal and other dimensions to facilitate the development and interoperability of tools build for it. Additional features are name-based addressing of data and internal consistency checks (e.g. checking for the right data order in calculations).

Depends R(>= 2.10.0), methods,

Imports stats, sp, maptools, abind, data.table, forcats

Suggests testthat, knitr, rmarkdown, reshape2, data.tree, raster, ncd4, covr

URL https://github.com/pik-piam/magclass,
     https://doi.org/10.5281/zenodo.1158580

BugReports https://github.com/pik-piam/magclass/issues

License LGPL-3 | file LICENSE

LazyData true

Encoding UTF-8

RoxygenNote 7.1.2

VignetteBuilder knitr

NeedsCompilation no

Author Jan Philipp Dietrich [aut, cre],
  Benjamin Leon Bodirsky [aut],
  Markus Bonsch [aut],
  Florian Humpenoeder [aut],
  Stephen Bi [aut],
  Kristine Karstens [aut],
Debbora Leip [aut],
Lavinia Baumstark [ctb],
Christoph Bertram [ctb],
Anastasis Giannousakis [ctb],
David Klein [ctb],
Ina Neher [ctb],
Michaja Pehl [ctb],
Anselm Schultes [ctb],
Miodrag Stevanovic [ctb],
Xiaoxi Wang [ctb],
Felicitas Beier [ctb]

Maintainer Jan Philipp Dietrich <dietrich@pik-potsdam.de>
Repository CRAN
Date/Publication 2021-09-30 10:10:01 UTC

R topics documented:

magclass-package ........................................... 4
add_columns .................................................. 4
add_dimension ................................................ 5
as.array-methods ........................................... 6
as.data.frame-methods ...................................... 6
as.RasterBrick ............................................. 7
clean_magpie ................................................ 8
collapseDim ................................................. 9
collapseNames .............................................. 10
colSums-methods ........................................... 11
complete_magpie .......................................... 11
convergence ................................................. 12
copy.attributes ........................................... 13
copy.magpie ............................................... 14
dimCode .................................................... 15
dimExists .................................................. 16
dimOrder .................................................... 17
dimReduce .................................................. 18
dimSums ..................................................... 19
escapeRegex ............................................... 20
fulldim ..................................................... 20
getCells ................................................... 21
getComment ............................................... 22
getCoords .................................................. 23
getCPR ..................................................... 24
getDim ..................................................... 25
getItems ................................................... 26
getNames .................................................. 27
getRegionList ............................................. 28
getRegions ................................................ 29
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>getSets</td>
<td>30</td>
</tr>
<tr>
<td>getYears</td>
<td>31</td>
</tr>
<tr>
<td>hasCoords</td>
<td>32</td>
</tr>
<tr>
<td>hasSets</td>
<td>33</td>
</tr>
<tr>
<td>head.magpie</td>
<td>34</td>
</tr>
<tr>
<td>is.temporal</td>
<td>35</td>
</tr>
<tr>
<td>isYear</td>
<td>35</td>
</tr>
<tr>
<td>lowpass</td>
<td>36</td>
</tr>
<tr>
<td>magclassdata</td>
<td>37</td>
</tr>
<tr>
<td>magpie-class</td>
<td>37</td>
</tr>
<tr>
<td>magpiesort</td>
<td>39</td>
</tr>
<tr>
<td>magpie_expand</td>
<td>40</td>
</tr>
<tr>
<td>magpie_expand_dim</td>
<td>41</td>
</tr>
<tr>
<td>magapply</td>
<td>42</td>
</tr>
<tr>
<td>maxample</td>
<td>43</td>
</tr>
<tr>
<td>mbind</td>
<td>44</td>
</tr>
<tr>
<td>mcalc</td>
<td>45</td>
</tr>
<tr>
<td>mselect</td>
<td>46</td>
</tr>
<tr>
<td>ncells</td>
<td>47</td>
</tr>
<tr>
<td>ndim</td>
<td>48</td>
</tr>
<tr>
<td>new.magpie</td>
<td>48</td>
</tr>
<tr>
<td>population_magpie</td>
<td>50</td>
</tr>
<tr>
<td>print.magpie</td>
<td>50</td>
</tr>
<tr>
<td>read.magpie</td>
<td>51</td>
</tr>
<tr>
<td>read.report</td>
<td>53</td>
</tr>
<tr>
<td>replace_non_finite</td>
<td>55</td>
</tr>
<tr>
<td>round-methods</td>
<td>55</td>
</tr>
<tr>
<td>rowSums-methods</td>
<td>56</td>
</tr>
<tr>
<td>setItems</td>
<td>57</td>
</tr>
<tr>
<td>setNames-methods</td>
<td>58</td>
</tr>
<tr>
<td>sizeCheck</td>
<td>58</td>
</tr>
<tr>
<td>time_interpolate</td>
<td>59</td>
</tr>
<tr>
<td>unwrap</td>
<td>60</td>
</tr>
<tr>
<td>where</td>
<td>61</td>
</tr>
<tr>
<td>wrap</td>
<td>61</td>
</tr>
<tr>
<td>write.magpie</td>
<td>62</td>
</tr>
<tr>
<td>write.report</td>
<td>65</td>
</tr>
<tr>
<td>write.report2</td>
<td>66</td>
</tr>
</tbody>
</table>

**Index** | 67
magclass-package  

Data Class and Tools for Handling Spatial-Temporal Data

Description

Data class for increased interoperability working with spatial-temporal data together with corresponding functions and methods (conversions, basic calculations and basic data manipulation). The class distinguishes between spatial, temporal and other dimensions to facilitate the development and interoperability of tools build for it. Additional features are name-based addressing of data and internal consistency checks (e.g. checking for the right data order in calculations).

Author(s)

Maintainer: Jan Philipp Dietrich <dietrich@pik-potsdam.de>

add_columns

Description

Function adds new columns to the existing magpie object.

Usage

add_columns(x, addnm = "new", dim = 3.1, fill = NA)

Arguments

x  
MAgPIE object which should be extended.

addnm  
The new elements that should be added to the (sub)dimension

dim  
The (sub)dimension to be filled either identified via name or dimension code (see dimCode for more information)

fill  
fill value of length 1 for the newly added columns (NA by default)

Value

The extended MAgPIE object

Author(s)

Jan Philipp Dietrich, Benjamin Bodirsky

See Also

add_dimension, dimCode
Examples

```r
a <- maxample("animal")
a2 <- add_columns(a, addnm = c("horse", "magpie"), dim = "species", fill = 42)
getItems(a2, dim = 3)
getItems(a2, dim = 3, split = TRUE)
head(a2[, , "magpie"])
```

Description

Function adds a name dimension as dimension number "dim" with the name "add" with an empty data column with the name "nm".

Usage

```r
add_dimension(x, dim = 3.1, add = NULL, nm = "dummy")
```

Arguments

- `x`: MAgPIE object which should be extended.
- `dim`: The dimension number of the new dimension (e.g. 3.1)
- `add`: The name of the new dimension
- `nm`: The name of the first entry in dimension "add".

Value

The extended MAgPIE object

Author(s)

Jan Philipp Dietrich, Benjamin Bodirsky

See Also

`add_columns`, `mbind`

Examples

```r
a <- maxample("animal")
str(add_dimension(a, dim = 3.2))
str(add_dimension(a, dim = 2.3, nm = paste0("d", 1:3)))
```
### as.array-methods

**Description**

~~ Methods for function as.array ~~

**Usage**

```r
## S4 method for signature 'magpie'
as.array(x)
```

**Arguments**

- `x` object which should be converted to an array

**Methods**

- `list("signature(x = "ANY")")` standard as.array-method
- `list("signature(x = "magpie\")")` Conversion takes place just by removing MAgPIE-object specific elements

### as.data.frame-methods

**Description**

~~ Methods for function as.data.frame ~~

**Usage**

```r
## S4 method for signature 'magpie'
as.data.frame(x, rev = 1)
```

**Arguments**

- `x` A MAgPIE-object
- `rev` The revision of the algorithm that should be used for conversion. `rev=1` creates columns with the predefined names Cell, Region, Year, Data1, Data2,... and Value, `rev=2` uses the set names of the MAgPIE object for naming and adds an attribute "dimtype" to the data.frame which contains information about the types of the different columns (spatial, temporal, data or value).
Methods

\texttt{list("signature(x = \"magpie\")")} Conversion creates columns for Cell, Region, Year, Data1, Data2,... and Value

Examples

\begin{verbatim}
pop <- maxample("pop")
head(as.data.frame(pop))
head(as.data.frame(pop,rev=2))
\end{verbatim}

Description

Convert magclass object to a RasterBrick object

Usage

\texttt{as.RasterBrick(x, res = NULL)}

Arguments

\begin{itemize}
  \item \textbf{x} MAgPIE object
  \item \textbf{res} spatial data resolution. If not provided it will be guessed.
\end{itemize}

Value

A RasterBrick object

Author(s)

Jan Philipp Dietrich

See Also

\texttt{getCoords}

Examples

\begin{verbatim}
if (requireNamespace("raster", quietly = TRUE)) {
  r <- raster::brick(ncols = 360, nrows = 180, nl = 4)
  r[85:89, 176:179] <- (1:20 %*% t(1:4))
  names(r) <- c("y2000..bla", "y2001..bla", "y2000..blub", "y2001..blub")
  m <- as.magpie(r)
  r2 <- as.RasterBrick(m)
}
\end{verbatim}
clean_magpie  

**MAgPIE-Clean**

**Description**

Function cleans MAgPIE objects so that they follow some extended magpie object rules (currently it makes sure that the dimnames have names and removes cell numbers if it is purely regional data)

**Usage**

```r
clean_magpie(x, what = "all")
```

**Arguments**

- `x`  
  MAgPIE object which should be cleaned.

- `what`  
  term defining what type of cleaning should be performed. Current modes are "cells" (removes cell numbers if the data seems to be regional - this should be used carefully as it might remove cell numbers in some cases in which they should not be removed), "sets" (making sure that all dimensions have names), "items" (replace empty elements with single spaces " ") and "all" (performing all available cleaning methods)

**Value**

The eventually corrected MAgPIE object

**Author(s)**

Jan Philipp Dietrich

**See Also**

"magpie"

**Examples**

```r
pop <- maxample("pop")
a <- clean_magpie(pop)
```
collapseDim  

**Description**  
This function will remove names in the data dimension which are the same for each element (meaning that this data dimension contains exactly one element) or, if forced, remove any other subdimension. It is a generalized version of the function `collapseNames`.

**Usage**  
collapseDim(x, dim = NULL, keepdim = NULL)

**Arguments**  
x  
MAgPIE object

dim  
Either NULL, dimension code or name of dimension or a vector of these. If set to NULL all single entry subdimensions will be removed as they are irrelevant to uniquely identify a data element. If specified, only the specified subdimensions will be removed (See `dimCode` for more details how to specify a subdimension). CAUTION: The function also allows to specify subdimensions which are otherwise needed to clearly identify an entry. By removing these subdimensions duplicates in the data will be created potentially causing problems in the further use of the data set. Be careful in removing subdimensions.

keepdim  
(only considered if `dim` is not specified) Can be used to converse single element subdimension which otherwise would get deleted. If `dim` is specified this setting will not have any effect.

**Value**  
The provided MAgPIE object with collapsed dimensions

**Author(s)**  
Jan Philipp Dietrich

**See Also**  
`getItems "magpie"`

**Examples**  
x <- new.magpie(c("GLO.1", "GLO.2"), 2000, c("bla.a", "bla.b"))
collapseDim(x)
collapseDim(x, keepdim = 1:2)
collapseDim(x, dim = 1.1)
collapseDim(x, dim = 3.2)
collapseNames

Collapse dataset names

Description

This function has been superseded by collapseDim which is a more generalized version of this function. Please use this one instead!

Usage

collapseNames(x, collapsedim = NULL, preservedim = NULL)

Arguments

- `x`: MAgPIE object
- `collapsedim`: If you want to remove the names of particular dimensions provide the dimensions here. Since the function only works in the third dimension, you have to count from there on (e.g. dim = 3.2 refers to collapsedim = 2). Alternatively, you can also specify the name of the dimension. Default: NULL. CAUTION with parameter collapsedim! You could also force him to remove dimnames, which are NOT the same for each element and so create duplicates in dimnames.
- `preservedim`: If you want to remove the name of particular dimensions except some, you can specify the dimension(s) to preserve here. See collapsedim for naming convention. Note that preservedim will be ignored in the case, of a specified collapsedim

Details

This function will remove names in the data dimension which are the same for each element (meaning that this data dimension contains exactly one element)

Value

The provided MAgPIE object with collapsed names

Author(s)

Jan Philipp Dietrich, David Klein, Xiaoxi Wang

See Also

collapseDim, getItems, "magpie"
colSums-methods

~ Methods for Function colSums and colMeans ~

Description

~ Methods for function colSums and colMeans ~

Usage

## S4 method for signature 'magpie'
colSums(x, na.rm = FALSE, dims = 1, ...)

Arguments

x 
object on which calculation should be performed

na.rm 
logical. Should missing values (including NaN) be omitted from the calculations?

dims 
integer: Which dimensions are regarded as "rows" or "columns" to sum over. For row*, the sum or mean is over dimensions dims+1, ...; for col* it is over dimensions 1:dims.

... further arguments passed to other colSums/colMeans methods

Methods

list("signature(x = "ANY")") normal colSums and colMeans method
list("signature(x = "magpie")") classical method prepared to handle MAgPIE objects

complete_magpie

~ complete_magpie ~

Description

MAgPIE objects can be incomplete to reduce memory. This function blows up a magpie object to its real dimensions, so you can apply unwrap.

Usage

complete_magpie(x, fill = NA, dim = 3)

Arguments

x 
MAgPIE object which should be completed.

fill 
Value that shall be written into the missing entries

dim 
dimensions in which the completion should take place (1, 2 and/or 3). For full completion use 1:3
Value

The completed MAgPIE object

Author(s)

Jan Philipp Dietrich, Benjamin Bodirsky

See Also

add_dimension, clean_magpie

Examples

```r
pop <- maxample("pop")
complete_magpie(pop)

ani <- maxample("animal")
complete_magpie(ani)
```

Description

Cross-Fades the values of one MAGPIE object into the values of another over a certain time

Usage

```r
convergence(
  origin,
  aim,
  start_year = NULL,
  end_year = NULL,
  direction = NULL,
  type = "smooth",
  par = 1.5
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>origin</td>
<td>an object with one name-column</td>
</tr>
<tr>
<td>aim</td>
<td>Can be twofold: An magpie object or a numeric value.</td>
</tr>
<tr>
<td>start_year</td>
<td>year in which the convergence from origin to aim starts. If set to NULL the the first year of aim is used as start_year</td>
</tr>
<tr>
<td>end_year</td>
<td>year in which the convergence from origin to aim shall be (nearly) reached. If set to NULL the the last year of aim is used as end_year.</td>
</tr>
</tbody>
</table>
**Direction**

NULL, "up" or "down". NULL means normal convergence in both directions, "up" is only a convergence if origin<aim, "down" means only a convergence if origin>aim.

**Type**

"smooth", "s", "linear" or "decay". Describes the type of convergence: linear means a linear conversion, s is an s-curve which starts from origin in start_year and reaches aim precisely in end_year. After 50 percent of the convergence time, it reaches about the middle of the two values. Its based on the function min(1, pos^4/(0.07+pos^4)*1.07) smooth is a conversion based on the function x^3/(0.1+x^3). In the latter case only 90% of convergence will be reached in the end year, because full convergence is reached in infinity. decay is a conversion based on the function x/(1.5 + x)*2.5.

**Par**

Parameter value for convergence function; currently only used for type="decay".

**Value**

Returns a time-series with the same timesteps as origin, which linearly fades into the values of the aim object.

**Author(s)**

Benjamin Bodirsky, Jan Philipp Dietrich

**Examples**

```r
pop <- maxample("pop")
population <- add_columns(pop, "MIX")
population[, , "MIX"] <- convergence(population[, , "A2"], population[, , "B1"])
```

**Description**

This function copies attributes from one object and assigns them to another.

**Usage**

```r
copy.attributes(  
  from,  
  to,  
  delete = c("names", "row.names", "class", "dim", "dimnames"),  
  delete2 = NULL  
)
```

```r
copy.attributes(  
  to,  
  
)```
```r
delete = c("names", "row.names", "class", "dim", "dimnames"),
delete2 = NULL
) <- value
Arguments
from          object from which the attributes should be taken
to            object to which the attributes should be written
delete        attributes which should not be copied. By default this are class specific attributes
               which might cause problems if copied to another object. But you can add or
               remove attributes from the vector.
delete2       Identical to delete and just added for convenience for the case that you want to
               delete additional attributes but do not want to repeat the vector given in delete.
               In the function both vectors, delete and delete2, are just merged to one deletion
               vector.
value         Same as "from" (object from which the attributes should be taken)
Functions
  • copy.attributes<- : assign attributes from object "value"
Author(s)
  Jan Philipp Dietrich
Examples

from <- array(12)
attr(from,"blablub") <- "I am an attribute!"
attr(from,"blablub2") <- "I am another attribute!"

print(attributes(from))

to <- as.magpie(0)
print(attributes(to))

copy.attributes(to) <- from
print(attributes(to))
```

---

**copy.magpie**

*Copy MAgPIE-files*

**Description**

This function copies MAgPIE-files from one location to another. During the copying it is also possible to change the file type (e.g. from ‘mz’ to ‘csv’).
dimCode

Usage

copy.magpie(input_file, output_file, round = NULL)

Arguments

input_file file, that should be copied
output_file copy destination
round number of digits the values should be rounded. NULL means no rounding

Author(s)

Jan Philipp Dietrich

See Also

read.magpie, write.magpie

Examples

# copy.magpie("bla.csv","blub.mz")

---

dimCode

dimCode

Description

Function converts a dimension name or number to a dimension Code used for MAgPIE objects

Usage

dimCode(dim, x, missing = 0, strict = FALSE, sep = ".")

Arguments

dim A vector of dimension numbers or dimension names which should be translated
x MAgPIE object in which the dimensions should be searched for.
missing Either a value to which a dimension should be set in case that it is not found (default is 0), or "stop" indicating that the function should throw an error in these cases.
strict if set to TRUE also properly set dimension names which refer to non-existing subdimensions will be treated as missing, otherwise these dimension codes will be returned, even if the subdimension does not exist
sep A character separating joined dimension names
Value

A dimension code identifying the dimension. Either a integer which represents the main dimensions (1=spatial, 2=temporal, 3=data) or a numeric, representing the subdimensions of a dimension (e.g. 3.2 for the second data dimension).

Author(s)

Jan Philipp Dietrich, Kristine Karstens

See Also

mselect, getDim

Examples

```r
pop <- maxample("pop")
dimCode(c("t", "scenario", "blablub"), pop)
```

dimExists

Description

Function checks whether a dimension exist in a MAgPIE objects

Usage

```r
dimExists(dim, x, sep = ".")
```

Arguments

- **dim**: A vector of dimension numbers or dimension names which should be checked for
- **x**: MAgPIE object in which the dimensions should be searched for.
- **sep**: A character separating joined dimension names

Value

Boolean indicating whether dimension exists or not

Author(s)

Jan Philipp Dietrich

See Also

dimCode
**Examples**

```r
pop <- maxample("pop")
dimExists(c("t", "scenario", "blablub"), pop)
```

---

**Description**

Changes the order of the sub-dimension in a magpie object similar to unwrapping and applying the `aperm` command, but more efficient.

**Usage**

```r
dimOrder(x, perm, dim = 3)
```

**Arguments**

- `x`: magpie object
- `perm`: vector with the new order of the sub-dimension. Missing sub-dimensions will be added automatically at the end
- `dim`: main dimension in which the order of sub-dimensions should be changed (1, 2 or 3)

**Value**

magpie object

**Author(s)**

Jan Philipp Dietrich, Benjamin Leon Bodirsky

**Examples**

```r
a <- maxample("animal")
head(a)
head(dimOrder(a, perm = 3:1, dim = 1))
head(dimOrder(a, perm = c(2,1,3), dim = 3))
```
**dimReduce**

**Description**

Remove dimensions which contain identical data for all elements in it

**Usage**

```
dimReduce(x, dim_exclude = NULL)
```

**Arguments**

- `x` MAgPIE object which should be reduced
- `dim_exclude` Vector with names of dimensions which must not be reduced

**Value**

The reduced MAgPIE object

**Author(s)**

Jan Philipp Dietrich

**See Also**

`add_dimension`

**Examples**

```r
# create data with 5 identical scenarios
p <- add_dimension(maxample("pop"), nm = paste0("scen", 1:5))
p
dimReduce(p)

# set years to same value
p[, , ] <- setYears(p[, 1, ], NULL)
p
dimReduce(p)

# set regions to same value
p[, , ] <- setCells(p[, , ], "GLO")
p
dimReduce(p)
```
Description

This function sums over any (sub-)dimension of a magpie object

Usage

`dimSums(x, dim = 3, na.rm = FALSE)`

Arguments

- `x` A MAgPIE-object
- `dim` The dimensions(s) to sum over. A vector of dimension codes or dimension names. See `dimCode` for more information
- `na.rm` logical. Should missing values (including NaN) be omitted from the calculations?

Value

A MAgPIE object with values summed over the specified dimensions

Author(s)

Jan Philipp Dietrich

See Also

`rowSums, getItems, dimCode`

Examples

```r
a <- maxample("animal")
dimSums(a, dim = c(1, 2, 3.2))
dimSums(a, dim = c("x", "y", "cell", "month"))
```
**escapeRegex**

**Description**

Escapes all symbols in a string which have a special meaning in regular expressions.

**Usage**

```r
escapeRegex(x)
```

**Arguments**

- `x` String or vector of strings that should be escaped.

**Value**

The escaped strings.

**Author(s)**

Jan Philipp Dietrich

**See Also**

`grep`

---

**fulldim**

**Reconstructs full dimensionality of MAgPIE objects**

**Description**

If a MAgPIE object is created from a source with more than one data dimension, these data dimensions are combined to a single dimension. fulldim reconstructs the original dimensionality and reports it.

**Usage**

```r
fulldim(x, sep = "." )
```

**Arguments**

- `x` A MAgPIE-object
- `sep` A character separating joined dimension names
getCells

Value
A list containing in the first element the dim output and in the second element the dimnames output of the reconstructed array.

Author(s)
Jan Philipp Dietrich

See Also
as.magpie, unwrap, wrap

Discussion
Extracts cell names of a MAgPIE-object

Usage
getCells(x)
getCells(x) <- value
setCells(object, nm = "GLO")

Arguments
x, object MAgPIE object
value, nm cell names the data should be set to.

Details
setCells is a shortcut to use a MAgPIE object with manipulated cell names. setCells uses the variable names "object" and "nm" in order to be consistent to the already existing function setNames.

Value
getCells returns cell names of the MAgPIE-object, whereas setCells returns the MAgPIE object with the manipulated cell names.

Functions
• getCells<-: set cell names
• setCells: set cell names
getComment

Author(s)
Jan Philipp Dietrich

See Also
getRegions, getNames, setNames, getCPR, read.magpie, write.magpie, "magpie"

Examples

```r
a <- as.magpie(1)
getCells(a)
setCells(a, "AFR")
```

description

Extracts the comment from a MAgPIE-object

Usage

```r
getComment(x)
getComment(x) <- value
setComment(object, nm = NULL)
```

Arguments

- **x, object**  MAgPIE object
- **value, nm**  A vector containing the comment.

Value

getComment returns the comment attached to a MAgPIE-object, NULL if no comment is present.
setComment returns the magpie object with the modified comment.

Functions

- `getComment<-`: set comment
- `setComment`: set comment

Author(s)
Markus Bonsch
getCoords

See Also

getRegions, getNames, getYears, getCPR, read.magpie, write.magpie, "magpie"

Examples

a <- as.magpie(1)
#returns NULL
gComment(a)
#set the comment
gComment(a)<-c("bla","blubb")
gComment(a)

getCoords

Get Coordinates

Description

Extracts spatial coordinates of a MAgPIE-object

Usage

getCoords(x, xlab = "x", ylab = "y")

getCoords(x, xlab = "x", ylab = "y") <- value

Arguments

x MAgPIE object
xlab label of x-dimension
ylab label of y-dimension
value coordinates as two column data.frame the data should be set to (first column = x, second column = y).

Value

coordinates of the MAgPIE-object

Functions

• getCoords<-: set coordinates

Author(s)

Jan Philipp Dietrich
getCPR

Description
Counts how often each element of the provided subdimension exists in the given data set. Originally created to count the number of cells in a region (this is also where its name originates from) it can now be used to count elements of any subdimension via the dim argument.

Usage
```r
getcPR(x, dim = 1.1)
```

Arguments
- `x` MAgPIE object or a resolution written as numeric (currently only data for 0.5 degree resolution is available).
- `dim` Dimension for which the items should be returned. Either number or name of dimension or a vector of these (in case of a vector all subdimensions must belong to the same main dimension!). See `dimCode` for more details.

Value
- cells per region

Author(s)
Jan Philipp Dietrich

See Also
- `getRegions`, `read.magpie`, `write.magpie`
**getDim**

Examples

```r
c getCPR(0.5)
a <- maxample("animal")
c getCPR(a, dim = "color")
c getCPR(a, dim = 3.2)
c getCPR(a, dim = "country")
c getCPR(a, dim = c("color", "species"))
```

---

**getDim**

---

Description

Function which tries to detect the dimension to which the given elems belong

Usage

```r
getDim(elems, x, fullmatch = FALSE, dimCode = TRUE)
```

Arguments

- `elems`: A vector of characters containing the elements that should be found in the MAgPIE object
- `x`: MAgPIE object in which elems should be searched for.
- `fullmatch`: If enabled, only dimensions which match exactly the elements provided will be returned. Otherwise, it is sufficient if elems contains a subset of the dimension.
- `dimCode`: If enabled, the dimCode will be returned, otherwise the name of the dimension.

Value

The name or dimCode of the dimensions in which elems were found.

Author(s)

Jan Philipp Dietrich

See Also

`mcalc`, `dimCode`

Examples

```r
pop <- maxample("pop")
getDim(c("AFR","CPA"),pop)
getDim(c("AFR","CPA"),pop,fullmatch=TRUE)
getDim(c("AFR","CPA"),pop,dimCode=FALSE)
```
**getItems**

### Description

Extract items of a given (sub-)dimension of a MAgPIE-object

### Usage

```r
getItems(x, dim = NULL, split = FALSE, full = FALSE)

getItems(x, dim, full = NULL, maindim = NULL, raw = FALSE) <- value
```

### Arguments

- **x**
  - MAgPIE object
- **dim**
  - Dimension for which the items should be returned. Either number or name of dimension or a vector of these. See `dimCode` for more details.
- **split**
  - Boolean which determines whether a main dimension should be split in subdimensions. Only applicable to main dimensions (1,2,3) and ignored for all other.
- **full**
  - if TRUE dimension names are returned as they are (including repetitions), if FALSE only the dimension elements (unique list of entries) are returned.
- **maindim**
  - main dimension the data should be added to (does not need to be set if `dim` exists in the data. Should be set if `dim` might not exist, or if `dim` might potentially exist in a different main dimension than the one anticipated).
- **raw**
  - if set to FALSE inputs will be corrected (e.g. dots replaced by the letter "p") if necessary. If TRUE data will be written as is (risking the creation of inconsistent objects).
- **value**
  - a vector with the length of the main dimension the dimnames should be replaced in / added to. If set to NULL the corresponding dimension will be removed.

### Value

items of the requested dimension in the MAgPIE-object. If `split=TRUE` and applied to a main dimension (1,2,3) a list of items for each sub-dimension.

### Functions

- `getItems<-`: set dimension names

### Author(s)

Jan Philipp Dietrich

### See Also

- `dimCode`
getNames

Examples

```r
x <- maxample("pop")
getItems(x, "scenario")
getItems(x, 3.1)
getItems(x, "i") <- paste0("REG", seq_len(ncells(x)))
getItems(x, "i")
y <- x[, 1, ]
getItems(y, "t") <- NULL
```

---

**getNames**

*Get dataset names*

Description

Extracts dataset names of a MAgPIE-object

Usage

```r
getNames(x, fulldim = FALSE, dim = NULL)
getNames(x, dim = NULL) <- value
```

Arguments

- **x**: MAgPIE object
- **fulldim**: specifies, how the object is treated. In case of FALSE, it is assumed that x is 3 dimensional and dimnames(x)[[3]] is returned. In case of TRUE, the dimnames of the real third dimension names are returned.
- **dim**: Argument to choose a specific data dimension either by name of the dimension or by number of the data dimension.
- **value**: a vector of names current names should be replaced with. If only one data element exists you can also set the name to NULL.

Details

setNames is a shortcut to use a MAgPIE object with manipulated data names. The setNames method uses the variable names "object" and "nm" in order to be consistent to the already existing function setNames.

Value

getNames returns data names of the MAgPIE-object, whereas setNames returns the MAgPIE object with the manipulated data names.

Functions

- `getNames<-`: set names
getRegionList

Get a list of cellular region-belongings

Description

Extracts a vector containing the region of each cell of a MAgPIE-object

Usage

getRegionList(x)

getRegionList(x) <- value

Arguments

x MAgPIE object

value A vector with ncell elements containing the regions of each cell.

Value

A vector with ncell elements containing the region of each cell.

Functions

* getRegionList<-: set region names
getRegions

Author(s)
Jan Philipp Dietrich

See Also
getRegions, getYears, getNames, getCPR, read.magpie, write.magpie, "magpie"

Examples

```r
# a <- read.magpie("example.mz")
# getRegionList(a)
```

Description
Extracts regions of a MAgPIE-object

Usage

```
getRegions(x)
getRegions(x) <- value
```

Arguments

- `x`: MAgPIE object
- `value`: Vector containing the new region names of the MAgPIE objects. If you also want to change the mapping of regions to cell please use `getRegionList` instead.

Value
Regions of the MAgPIE-object

Functions

- `getRegions<-`: overwrite region names

Author(s)
Jan Philipp Dietrich

See Also
getYears, getNames, getCPR, read.magpie, write.magpie, "magpie"
Examples

```r
# a <- read.magpie("example.mz")
# getRegions(a)
```

### Description

Extracts sets of a MAgPIE-object if available

### Usage

```r
getSets(x, fulldim = TRUE, sep = ".")
getSets(x, fulldim = TRUE, sep = ".") <- value
```

### Arguments

- `x` MAgPIE object
- `fulldim` bool: Consider dimension 3 as a possible aggregate of more dimensions (TRUE) or stick to it as one dimension (FALSE)
- `sep` A character separating joined dimension names
- `value` A vector with set names you want to replace the current set names of the object with.

### Value

Sets of the MAgPIE-object. If no information about contained sets is available NULL

### Functions

- `getSets<-`: replace set names

### Author(s)

Markus Bonsch, Jan Philipp Dietrich

### See Also

`getRegions`, `getNames`, `getYears`, `getCPR`, `read.magpie`, `write.magpie`, "magpie"
`getYears`  

**Examples**

```r
a <- new.magpie("GLO.1", 2000, c("a.o1", "b.o1", "a.o2"))
getSets(a) <- c("reg", "cell", "t", "bla", "blub")
getSets(a)

getSets(a)["d3.1"] <- "BLA"
getSets(a, fulldim = FALSE)
getSets(a)
```

**getYears**  

*Get years*

**Description**

Extracts years of a MAgPIE-object

**Usage**

```r
getYears(x, as.integer = FALSE)
getYears(x) <- value
setYears(object, nm = NULL)
```

**Arguments**

- `x, object`  
  MAgPIE object  
- `as.integer`  
  Switch to decide, if output should be the used year-name (e.g. "y1995") or the year as integer value (e.g. 1995)
- `value, nm`  
  Years the data should be set to. Either supplied as a vector of integers or a vector of characters in the predefined year format ("y0000"). If only 1 year exist you can also set the name of the year to NULL.

**Details**

`setYears` is a shortcut to use a MAgPIE object with manipulated year names. `setYears` uses the variable names "object" and "nm" in order to be consistent to the already existing function `setNames`.

**Value**

`getYears` returns years of the MAgPIE-object, whereas `setYears` returns the MAgPIE object with the manipulated years.

**Functions**

- `getYears<-` : rename years
- `setYears` : set years
Author(s)
Jan Philipp Dietrich

See Also
getRegions, getNames, setNames, getCPR, read.magpie, write.magpie, "magpie"

Examples

```r
a <- as.magpie(1)
getYears(a)
setYears(a, 1995)
```

---

### hasCoords

**Has Coordinates**

Checks, whether object contains coordinates.

#### Usage

```r
hasCoords(x, xlab = "x", ylab = "y")
```

#### Arguments

- **x**: MAgPIE object
- **xlab**: label of x-dimension
- **ylab**: label of y-dimension

#### Value

Boolean indicating whether coordinates were found or not

Author(s)
Jan Philipp Dietrich

See Also
getCoords
hasSets

Examples

hasCoords(maxample("pop"))
hasCoords(maxample("animal"))

hasSets 

Has Sets

Description

Checks, whether set names have been set

Usage

hasSets(x)

Arguments

x MAgPIE object

Value

Boolean indicating whether coordinates were found or not

Author(s)

Jan Philipp Dietrich

See Also

getCoords

Examples

hasSets(maxample("pop"))
hasSets(maxample("animal"))
Description

head and tail methods for MAgPIE objects to extract the head or tail of an object

Usage

```r
## S3 method for class 'magpie'
head(x, n1 = 3L, n2 = 6L, n3 = 2L, ...)
```

Arguments

- `x`: MAgPIE object
- `n1`, `n2`, `n3`: number of lines in first, second and third dimension that should be returned. If the given number is higher than the length of the dimension all entries in this dimension will be returned.
- `...`: arguments to be passed to or from other methods.

Value

head returns the first `n1` x `n2` x `n3` entries, tail returns the last `n1` x `n2` x `n3` entries.

Author(s)

Jan Philipp Dietrich

See Also

`head`, `tail`

Examples

```r
pop <- maxample("pop")
head(pop)
tail(pop, 2, 4, 1)
```
is.temporal

Description
Functions to find out whether a vector consists of strings consistent with the definition for auto-detection of temporal or spatial data.

Usage
is.temporal(x)

Arguments
x A vector

Value
Returns TRUE or FALSE

Author(s)
Jan Philipp Dietrich

Examples
is.temporal(1991:1993)
is.spatial(c("GLO","AFR"))

isYear

Description
Function to find out whether a vector consists of strings in the format "yXXXX" or "XXXX" with X being a number

Usage
isYear(x, with_y = TRUE)

Arguments
x A vector
with_y indicates which dataformat years have to have (4-digit without y (e.g.1984) or 5digit including y (y1984))
lowpass

Value
Returns a vector of the length of x with TRUE and FALSE

Author(s)
Benjamin Bodirsky

Examples

```r
x <- c("1955", "y1853", "12a4")
isYear(x, with_y = TRUE)
isYear(x, with_y = FALSE)
```

---

lowpass  Lowpass Filter

Description
Filters high frequencies out of a time series. The filter has the structure \( x'(n) = \frac{x(n-1)+2x(n)+x(n+1)}{4} \)

Usage
```r
lowpass(x, i = 1, fix = NULL, altFilter = NULL, warn = TRUE)
```

Arguments

- **x**: Vector of data points, that should be filtered or MAgPIE object
- **i**: number of iterations the filter should be applied to the data
- **fix**: Fixes the starting and/or ending data point. Default value is NULL which doesn’t fix any point. Available options are: "start" for fixing the starting point, "end" for fixing the ending point and "both" for fixing both ends of the data.
- **altFilter**: set special filter rule to indexes defined in this parameter. The special filter has the structure \( x'(n) = \frac{2x(n)+x(n+1)}{3} \)
- **warn**: boolean deciding whether lowpass issues a warning for critical parameter choices or not

Value
The filtered data vector or MAgPIE object

Author(s)
Jan Philipp Dietrich, Misko Stevanovic
Examples

`lowpass(c(1, 2, 11, 3, 4))`
# to fix the starting point
`lowpass(c(0, 9, 1, 5, 14, 20, 6, 11, 0), i = 2, fix = "start")`
an argument tidy=TRUE can be used to indicate that the data.frame structure is following the rules of tidy data (last column is the data column all other columns contain dimension information). This information will help the conversion. sep defines the dimension separator (default is ".") and replacement defines how the separator as a reserved character should be converted in order to not mess up with the object (default "_"). Another available argument for conversions of data.frames and quite objects to magpie is filter if set to TRUE (default) "." (separator) will be replaced with the replacement character and empty entries will be replaced with a single space. If set to FALSE no filter will be applied to the data.

**Objects from the Class**

Objects can be created by calls of the form `new("magpie", data, dim, dimnames, ...)`. MAgPIE objects have three dimensions (cells,years,datatype) and the dimensionnames of the first dimension have the structure "REGION.cellnumber". MAgPIE-objects behave the same like array-objects with 2 exceptions:

1. Dimensions of the object will not collapse (e.g. `x[1,1,1]` will remain 3D instead of becoming 1D)
2. It is possible to extract full regions just by typing `x["REGIONNAME",,].`

Please mind following standards:

- Header must not contain any purely numeric entries, but combinations of characters and numbers are allowed (e.g. "bla","12" is forbidden, whereas "bla","b12" is allowed)
- Years always have the structure "y" + 4-digit number, e.g. "y1995"
- Regions always have the structure 3 capital letters, e.g. "AFR" or "GLO"

This standards are necessary to allow the scripts to detect headers, years and regions properly and to have a distinction to other data.

**Author(s)**

Jan Philipp Dietrich

**See Also**

`read.magpie`, `write.magpie`, `getRegions`, `getYears`, `getNames`, `getCPR`, `ncells`, `nyears`, `ndata`  

**Examples**

```r
showClass("magpie")

pop <- maxample("pop")

# returning PAO and PAS for 2025
pop["PA", 2025, , pmatch = "left"]

# returning CPA for 2025
```
mmagpiesort

Description

Brings the spatial and temporal structure of MAgPIE objects in the right order. This function is especially useful when you create new MAgPIE objects as the order typically should be correct for MAgPIE objects.

Usage

magpiesort(x)

Arguments

x MAgPIE object which might not be in the right order.
### Description

Expands a MAgPIE object based on a reference.

#### Usage

```r
magpie_expand(x, ref)
```

#### Arguments

- `x`: MAgPIE object that should be expanded.
- `ref`: MAgPIE object that serves as a reference.

#### Details

Expansion means here that the dimensions of `x` are expanded accordingly to `ref`. Please note that this is really only about expansion. In the case that one dimension of `ref` is smaller than of `x` nothing happens with this dimension. At the moment `magpie_expand` is only internally available in the `magclass` library.

You can influence the verbosity of this function by setting the option "magclass.verbosity". By default verbosity is set to 1 which means that only warnings are returned. Setting verbosity to 2 means that warnings as well as additional notes are returned. This is done by `options(verbosity.level=2)`.

With version 5 of the package `magpie_expand` has been updated to a newer version (currently 2.1) and since version 6 this is the only currently supported version. To switch to the old setup you have to install `magclass` in a version < 6 and set `options(magclass_expand_version=1)`.

By default expansion is based on the elements in a dimension ignoring the set name of the dimension. To expand based on set names instead of contents (recommended) you can switch `options(magclass_setMatching=TRUE)`.
**Description**

Expands a single MAgPIE object dimension

**Usage**

```r
magpie_expand_dim(x, ref, dim = 1)
```

**Arguments**

- `x`: MAgPIE object that should be expanded
- `ref`: MAgPIE object that serves as a reference
- `dim`: dimension that should be expanded

**Details**

Expansion means here that the dimensions of `x` are expanded accordingly to `ref`. Please note that this is really only about expansion. In the case that one dimension of `ref` is smaller than of `x` nothing happens with this dimension. At the moment `magpie_expand` is only internally available in the magclass library.

In contrast to `magpie_expand` this function is expanding only a single dimension. It is meant as a support function for `magpie_expand` itself.
Value
An expanded version of x.

Author(s)
Jan Philipp Dietrich

See Also
-as.magpie, options

Examples
```r
d <- new.magpie(c("AFR.BLUB.1", "AFR.BLUB.2", "EUR.BLUB.1", "AFR.BLA.1", "AFR.BLA.2", "EUR.BLA.1"), fill = 1)
getSets(d)[1:3] <- c("reg", "b", "i")
e <- new.magpie(c("BLA.AFR.A", "BLA.EUR.A", "BLUB.AFR.A", "BLUB.EUR.A", "BLA.AFR.B", "BLA.EUR.B", "BLUB.AFR.B", "BLUB.EUR.B"), fill = 2)
getSets(e)[1:3] <- c("b", "reg", "a")
magclass:::magpie_expand_dim(d, e, dim = 1)
```

Description
apply command for magpieobjects. Very efficient for replacing loops.

Usage
```r
magpply(X, FUN, MARGIN = NULL, DIM = NULL, ..., INTEGRATE = FALSE)
```

Arguments
- **X**: magpie object
- **FUN**: function that shall be applied X
- **MARGIN**: dimension over which FUN shall be applied (like a loop over that dimension). This dimension will be preserved in the output object (see also DIM).
- **DIM**: dimension in which FUN shall be applied. This dimension will be missing in the output. MARGIN and DIM are opposite ways of expressing the dimensions to be addressed and you must only use one of them with MARGIN excluding dimensions from the calculation and DIM including them.
- **...**: further parameters passed on to FUN
- **INTEGRATE**: if TRUE, the output will be filled into an magpie object of the same dimensionality as X
**maxample**

**Value**

magpie object

**Author(s)**

Jan Philipp Dietrich, Benjamin Leon Bodirsky

**Examples**

```r
pop <- maxample("pop")
magapply(pop, FUN = sum, MARGIN = 2)
fourdim <- pop * setNames(pop, c("jkk", "lk"))
magapply(fourdim, FUN = sum, MARGIN = c(1, 3.1))
```

**Description**

A collection of magclass example data sets

**Usage**

```r
maxample(data)
```

**Arguments**

```r
data
```

name of the example data set. Currently available are "pop" (regional population data, previously named "population_magpie") and "animal" (fictional, high-dimensional animal sighting data set).

**Value**

the chosen example data set

**Author(s)**

Jan Philipp Dietrich

**Examples**

```r
p <- maxample("pop")
str(p)

a <- maxample("animal")
str(a)
getItems(a, split = TRUE)
```
Description

Merges MAgPIE-objects with identical structure in two dimensions. If data differs in the temporal or spatial dimension each year or region/cell must appear only once!

Usage

mbind(...)

Arguments

... MAgPIE objects or a list of MAgPIE objects that should be merged.

Value

The merged MAgPIE object

Author(s)

Jan Philipp Dietrich, Misko Stevanovic

See Also

"magpie"

Examples

m <- new.magpie(c("AFR", "CPA", "EUR"), c(1995, 2005), "Data1", fill = c(1, 2, 3, 4, 5, 6))
ms <- dimSums(m, dim = 3.1)
mbind(m, ms)
my <- new.magpie(getRegions(m), 2010, getNames(m), fill = c(6, 6, 4))
mbind(m, my)
md <- new.magpie(getRegions(m), getYears(m), "Data2", fill = c(7, 6, 5, 7, 8, 9))
mbind(m, md)

pop <- maxample("pop")
a <- mbind(pop, pop)
dim(pop)
dim(a)
Description

Select values from a MAgPIE-object

Usage

mcalc(x, f, dim = NULL, append = FALSE)

Arguments

x  
MAgPIE object

f  
A formula describing the calculation that should be performed

dim  
The dimension in which the manipulation should take place. If set to NULL
function tries to detect the dimension automatically.

append  
If set to TRUE the result will be appended to x, otherwise the result will be
returned.

Details

This functions only work for MAgPIE objects with named dimensions as the dimension name
(set_name) has to be used to indicate in which dimension the entries should be searched for!

Value

The calculated MAgPIE object in the case that append is set to FALSE. Otherwise nothing is re-
turned (as x is appended in place)

Author(s)

Jan Philipp Dietrich

See Also

mselect

Examples

pop <- maxample("pop")
pop
mcalc(pop, X12 ~ A2 * B1, append = TRUE)
pop
mcalc(pop, `Nearly B1` ~ 0.5 * A2 + 99.5 * B1)
Description

Select values from a MAgPIE-object

Usage

mselect(x, ..., collapseNames = FALSE)

mselect(x, ...) <- value

Arguments

x

MAgPIE object

... entry selections of the form set_name=c(set_elem1,set_elem2). Alternatively a single list element containing these selections can be provided.

collapseNames

Boolean which decides whether names should be collapsed or not.

value values on which the selected magpie entries should be set.

Details

This functions only work for MAgPIE objects with named dimensions as the dimension name (set_name) has to be used to indicate in which dimension the entries should be searched for!

Value

The reduced MAgPIE object containing only the selected entries or the full MAgPIE object in which a selection of entries was manipulated.

Functions

• mselect<-. replace values in magpie object

Author(s)

Jan Philipp Dietrich

See Also

collapseNames, "magpie"

Examples

pop <- maxample("pop")
mselect(pop, i = c("AFR", "EUR"), scenario = "A2", t = "y2035")
ncells

Count elements

Description
Functions to count the number of cells/years/datasets/regions of an MAgPIE-object

Usage
ncells(x)
ndata(x)
nregions(x)
nyears(x)

Arguments
x

A MAgPIE-object

Value
value
The number of cells/years/datasets/regions of x

Functions
• ndata: count datasets
• nregions: count regions
• nyeats: count years

Author(s)
Jan Philipp Dietrich

Examples
a <- is.magpie(NULL)
ncells(a)
nyears(a)
data(a)
nregions(a)
ndim

*Count sub-dimensions*

**Description**

Functions to count the subdimensions of an MAgPIE-object

**Usage**

```r
ndim(x, dim = NULL)
```

**Arguments**

- **x**: A MAgPIE-object
- **dim**: main dimension in which the sub-dimensions should be counted. If NULL the sum of all subdimensions is returned

**Value**

Number of subdimensions

**Author(s)**

Jan Philipp Dietrich

**Examples**

```r
a <- maxample("animal")
ndim(a)
ndim(a,1)
ndim(a,2)
ndim(a,3)
```

---

**new.magpie**

*new.magpie*

**Description**

Creates a new MAgPIE object
new.magpie

Usage

new.magpie(
  cells_and_regions = "GLO",
  years = NULL,
  names = NULL,
  fill = NA,
  sort = FALSE,
  sets = NULL,
  unit = "unknown"
)

Arguments

cells_and_regions
  Either the region names (e.g. "AFR"), or the cells (e.g. 1:10), or both in combination (e.g. "AFR.1"). NULL means no spatial element.

years
  dimnames for years in the format "yXXXX" or as integers. NULL means one year which is not further specified

names
  dimnames for names. NULL means one data element which is not further specified

fill
  Default value for the MAgPIE object

sort
  Boolean. Decides, whether output should be sorted or not.

sets
  A vector of dimension names. See getSets for more information.

unit
  A character which sets the MAgPIE object's unit field in its metadata attribute

Value

an empty magpie object filled with fill, with the given dimnames

Author(s)

Benjamin Bodirsky, Jan Philipp Dietrich

See Also

as.magpie

Examples

a <- new.magpie(1:10, 1995:2000)
b <- new.magpie(c("AFR", "CPA"), "y1995", c("bla", "blub"), sets = c("i", "t", "value"))
c <- new.magpie()
Description

Example dataset for a regional MAgPIE object

Value

A2 and B1 population scenario from SRES

Author(s)

Benjamin Bodirsky

Description

print method for MAgPIE objects for convenient display of magpie data.

Usage

```r
## S3 method for class 'magpie'
print(x, drop = TRUE, reshape = FALSE, ...)
```

Arguments

- `x` : MAgPIE object
- `drop` : argument which controls whether empty dimensions should be skipped or not.
- `reshape` : argument that controls tabular representation of nested data dimension cross-tables, FALSE will reproduce standard print behavior any pair of two dimension numbers will create a table for these two dims, and loop over the other dimensions
- `...` : arguments to be passed to or from other methods.

Value

print displays the given MAgPIE object on screen.

Author(s)

Jan Philipp Dietrich, Kristine Karstens, Felicitas Beier
read.magpie

See Also
print

Examples

```
pop <- maxample("pop")
print(pop)
print(pop[, 1, 1], drop = FALSE)
print(pop[, 1, 1])
```

Description

Reads a MAgPIE-file and converts it to a 3D array of the structure (cells,years,datacolumn)

Usage

```
read.magpie(
  file_name,
  file_folder = "",
  file_type = NULL,
  as.array = FALSE,
  comment.char = "*",
  check.names = FALSE,
...)
```

Arguments

- `file_name` file name including file ending (wildcards are supported). Optionally also the full path can be specified here (instead of splitting it to `file\_name` and `file\_folder`)
- `file_folder` folder the file is located in (alternatively you can also specify the full path in `file\_name` - wildcards are supported)
- `file_type` format the data is stored in. Currently 13 formats are available: "rds" (recommended compressed format), "cs2" & "cs2b" (cellular standard MAgPIE format), "csv" (regional standard MAgPIE format), "cs3" (multidimensional format compatible to GAMS), "cs4" (alternative multidimensional format compatible to GAMS, in contrast to cs3 it can also handle sparse data), "csvr", "cs2r", "cs3r" and "cs4r" which are the same formats as the previous mentioned ones with the only difference that they have a REMIND compatible format, "m" (binary MAgPIE format "magpie"), "mz" (compressed binary MAgPIE format "magpie zipped") "put" (format used primarily for the REMIND-MAgPIE coupling) and "asc", (ASCII-Grid format as used by ArcGis). If `file\_type=NULL` the file ending of the `file\_name` is used as format. If format is different to the formats mentioned standard MAgPIE format is assumed.
Should the input be transformed to an array? This can be useful for regional or global inputs, but all advantages of the magpie-class are lost.

character: a character vector of length one containing a single character or an empty string. Use "" to turn off the interpretation of comments altogether. If a comment is found it will be stored in attr("comment"). In text files the comment has to be at the beginning of the file in order to be recognized by read.magpie.

logical. If TRUE then the names of the variables in the data frame are checked to ensure that they are syntactically valid variable names. Same functionality as in read.table.

additional arguments passed to specific read functions (e.g. varname for specifying the variable to be read in from a multi-variable NCDF file.)

This function reads from 13 different MAgPIE file types. "rds" is a R-default format for storing R objects. "cs2" or "cs2b" is the new standard format for cellular data with or without header and the first columns (year, regiospatial) or only (regiospatial), "csv" is the standard format for regional data with or without header and the first columns (year, region, cellnumber) or only (region, cellnumber). "cs3" is a format similar to csv and cs2, but with the difference that it supports multidimensional data in a format which can be read by GAMS. "put" is a newly supported format which is mostly used for the REMIND-MAgPIE coupling. This format is only partly supported at the moment. "asc" is the AsciiGrid format (for example used for Arc Gis data). "nc" is the netCDF format (only "nc" files written by write.magpie can be read). All these variants are read without further specification. "magpie" (.m) and "magpie zipped" (.mz) are new formats developed to allow a less storage intensive management of MAgPIE-data. The only difference between both formats is that .mz is gzipped whereas .m is not compressed. So .mz needs less memory, whereas .m might have a higher compatibility to other languages.

Since library version 1.4 read.magpie can also read regional or global MAgPIE csv-files.

The binary MAgPIE formats .m and .mz have the following content/structure (you only have to care for that if you want to implement read.magpie/write.magpie functions in other languages):

- FileFormatVersion (Current file format version number (currently 6) | integer | 2 Byte)
- ncharComment (Number of character bytes of the file comment | integer | 4 Byte)
- nbyteMetadata (Number of bytes of the serialized metadata | integer | 4 Byte)
- ncharSets (Number of characters bytes of all regionnames + 2 delimiter | integer | 2 Byte)
- nyears (Number of years | integer | 2 Byte)
- yearList (All years of the dataset (0, if year is not present) | integer | 2*nyears Byte)
- ncells (Number of cells | integer | 4 Byte)
- nchar_cell (Number of characters bytes of all regionnames + (nreg-1) for delimiters | integer | 4 Byte)
The `read.report` function reads the content of a reporting file (a file in the model intercomparison file format *.mif) into a list of MAgPIE objects or a single MAgPIE object.

**Usage**

```r
read.report(file, as.list = TRUE)
```

**Arguments**

- `file` : file name the object should be read from.
- `as.list` : if TRUE a list is returned (default), if FALSE it is tried to merge all information in one MAgPIE object (still under development and works currently only if the entries for the different models and scenarios have exactly the same regions and years).

**Details**

The **Model Intercomparison File Format (MIF)** is the default file format for data produced by Integrated Assessment Models. It is based on the common format used for Model Intercomparison Projects such as EMF and SSP with some slight changes/clarifications in its definition. For interactions between models this format should be used. For everything else it is at least recommended to use this format, too.
Aim of this standardization is to achieve a more flexible and smooth communication between models and to facilitate the creation of aggregated outputs from integrated assessment scenario runs which then can easily be uploaded to external databases such as the EMF or SSP database. By using this standard most of the required decisions for a working input output interface between models have already been specified which significantly reduces the required work to get a new interaction running.

**Definition**

The format is characterized by the following features:

- The file ending is ".mif"
- The file is written in ASCII format
- Entries are separated with ";", every line ends with a ";"
- The file always contains a header
- The format of the header is: Model;Scenario;Region;Variable;Unit;<ADDITIONAL_COLUMNS>;<YEARS>;

The first 5 entries always have to exist, <ADDITIONAL_COLUMNS> is additional information which can be added optionally (e.g. "Description") and <YEARS> are the years for which data is delivered. <YEARS> are always written as 4 digit numbers. In the (very unlikely) case that a year before 1000 is used the number has to start with a 0, e.g. 0950. <ADDITIONAL_COLUMNS> can be anything, there are no further rules at the moment what it can contain. However, there are strict rules for naming these columns. Allowed are single names starting with a capital letter without special characters in it except "_" which is allowed. Examples: "Description" allowed, "More Description" not allowed, "More_Description" allowed, "123Description" not allowed, "Description123" allowed. Scripts using this format must be able to ignore additional columns. For years there are no specific limitations/requirements which years should be reported. Scripts dealing with this data must be able to work with different temporal resolutions. For variables basically everything can be reported here. Missing values have to be marked with "N/A".

**Author(s)**

Jan Philipp Dietrich

**See Also**

write.report

**Examples**

```r
## Not run:
read.report("report.csv")

## End(Not run)
```
replace_non_finite  Replace Non-Finite Data

Description

Replaces all instances of non-finite data (NA, NaN, Inf, and -Inf).

Usage

replace_non_finite(x, replace = 0)

Arguments

x  A vector or magpie object.
replace  A value to replace non-finite data with.

Value

A vector or magpie object, same as x.

Author(s)

Michaja Pehl

Examples

part <- new.magpie(letters[1:3], years = 'y1995', names = 'foo')
total <- new.magpie(letters[1:3], years = 'y1995', names = 'foo')

part[, ,] <- c(0, 1, 2)
total[, ,] <- c(0, 10, 10)

part / total

replace_non_finite(part / total)

round-methods  Round-method for MAgPIE objects

Description

Round-method for MAgPIE-objects respectively. Works exactly as for arrays.
Usage

## S4 method for signature 'magpie'
round(x, digits = 0)

Arguments

x a magpie object
digits integer indicating the number of decimal places (round) or significant digits (sig-
nif) to be used. Negative values are allowed.

Methods

x = "magpie" works as round(x) for arrays.

rowSums-methods

Description

~~ Methods for function rowSums and rowMeans ~~

Usage

## S4 method for signature 'magpie'
rowSums(x, na.rm = FALSE, dims = 1, ...)

Arguments

x object on which calculation should be performed
na.rm logical. Should missing values (including NaN) be omitted from the calcula-
tions?
dims integer: Which dimensions are regarded as "rows" or "columns" to sum over. For row*, the sum or mean is over dimensions dims+1, ...; for col* it is over dimensions 1:dims.
...
... further arguments passed to other colSums/colMeans methods

Methods

list("signature(x = \"ANY\")") normal rowSums and rowMeans method
list("signature(x = \"magpie\")") classical method prepared to handle MAgPIE objects
Description

Set items of a given (sub-)dimension of a MAgPIE-object

Usage

setItems(x, dim, value, maindim = NULL, raw = FALSE)

Arguments

x  
MAgPIE object

dim  
Dimension for which the items should be returned. Either number or name of
dimension or a vector of these. See dimCode for more details.

value  
a vector with the length of the main dimension the dimnames should be replaced
in / added to. If set to NULL the corresponding dimension will be removed.

maindim  
main dimension the data should be added to (does not need to be set if dim exists
in the data. Should be set if dim might not exist, or if dim might potentially exist
in a different main dimension than the one anticipated).

raw  
if set to FALSE inputs will be corrected (e.g. dots replaced by the letter "p") if
necessary. If TRUE data will be written as is (risking the creation of inconsistent
objects).

Value

the manipulated MAgPIE object

Author(s)

Jan Philipp Dietrich

See Also

getItems

Examples

x <- maxample("pop")
setItems(x, "i", paste0("REG", 1:ncells(x)))
setNames-methods  

Get dataset names

Description

Extracts dataset names of a MAgPIE-object

Usage

## S4 method for signature 'magpie'
setNames(object = nm, nm)

Arguments

- **object**  
  MAgPIE object

- **nm**  
  a vector of names current names should be replaced with. If only one data element exists you can also set the name to NULL.

Details

setNames is a shortcut to use a MAgPIE object with manipulated data names. The setNames method uses the variable names "object" and "nm" in order to be consistent to the already existing function setNames.

Methods

- list("signature(object = "ANY")") normal setNames method
- list("signature(object = "magpie")") setNames for MAgPIE objects

See Also

- getNames,

sizeCheck

sizeCheck

Description

Calculates expected magclass object length and checks that it stays below the limit defined with magclass_sizeLimit (default = 10^9). This is useful to prevent out of memory errors in case of unwanted object expansions Ignored if getOption("magclass_sizeLimit") is negative.

Usage

sizeCheck(dim)
Arguments

`dim` dimensions of the current object as returned by function `dim`

Author(s)

Jan Philipp Dietrich

Examples

```r
pop <- maxample("pop")
magclass::sizeCheck(dim(pop))
```

Description

Function to extrapolate missing years in MAgPIE objects.

Usage

```r
time_interpolate(
  dataset,
  interpolated_year,
  integrate_interpolated_years = FALSE,
  extrapolation_type = "linear"
)
```

Arguments

- `dataset` An MAgPIE object
- `interpolated_year` Vector of years, of which values are required. Can be in the formats 1999 or y1999.
- `integrate_interpolated_years` FALSE returns only the dataset of the interpolated year, TRUE returns the whole dataset, including all years of data and the interpolated year
- `extrapolation_type` Determines what happens if extrapolation is required, i.e. if a requested year lies outside the range of years in dataset. Specify "linear" for a linear extrapolation. "constant" uses the value from dataset closest in time to the requested year.

Value

Uses linear extrapolation to estimate the values of the interpolated year, using the values of the two surrounding years. If the value is before or after the years in data, the two closest neighbours are used for extrapolation.
unwrap

Author(s)
Benjamin Bodirsky, Jan Philipp Dietrich

See Also
convergence

Examples

```r
p <- maxample("pop")
time_interpolate(p, "y2000", integrate = TRUE)
time_interpolate(p, c("y1980", "y2000"), integrate = TRUE, extrapolation_type = "constant")
```

unwrap Unwrap

Description
Creates a higher dimensional array by separating all subdimensions in the third dimension of a MAgPIE object and returning them as separate dimension.

Usage
```r
unwrap(x, sep = NULL)
```

Arguments
- **x**: A MAgPIE object
- **sep**: deprecated, please do not use anymore

Value
An array with the full dimensionality of the original data

Author(s)
Jan Philipp Dietrich

See Also
wrap, fulldim

Examples
```r
a <- as.magpie(array(1:6, c(3, 2), list(c("bla", "blub", "ble"), c("up", "down"))))
unwrap(a)
```
**where**

**Description**

Analysis function for magpie objects

**Usage**

```r
where(x, plot = NULL)
```

**Arguments**

- `x`: A logical statement with a magpie object
- `plot`: deprecated. Use the function `whereplot` in package luplot.

**Value**

A list of analysis parameters

**Author(s)**

Benjamin Leon Bodirsky, Jan Philipp Dietrich

**See Also**

`whereplot` in package luplot

**Examples**

```r
p <- maxample("pop")
where(p > 500)
```

---

**wrap**

**Wrap**

**Description**

Reshape an array or a matrix by permuting and/or joining dimensions.

**Usage**

```r
wrap(x, map = list(NA), sep = ".")
```
Arguments

- x: An array
- map: A list of length equal to the number of dimensions in the reshaped array. Each element should be an integer vectors specifying the dimensions to be joined in corresponding new dimension. One element may equal NA to indicate that that dimension should be a join of all non-specified (remaining) dimensions. Default is to wrap everything into a vector.
- sep: A character separating joined dimension names

Note

This function is extracted from the R.utils library which is licensed under LGPL>=2.1 and written by Henrik Bengtsson.

Author(s)

Henrik Bengtsson, Jan Philipp Dietrich

See Also

unwrap.fulldim

write.magpie  Write MAgPIE-object to file

Description

Writes a MAgPIE-3D-array (cells,years,datacolumn) to a file in one of three MAgPIE formats (standard, "magpie", "magpie zipped")

Usage

write.magpie(
  x,
  file_name,
  file_folder = "",
  file_type = NULL,
  append = FALSE,
  comment = NULL,
  comment.char = "*",
  mode = NULL,
  ...
)

Arguments

- **x**: MAgPIE-object
- **file_name**: file name including file ending (wildcards are supported). Optionally also the full path can be specified here (instead of splitting it to `file\_name` and `file\_folder`)
- **file_folder**: folder the file should be written to (alternatively you can also specify the full path in `file\_name` - wildcards are supported)
- **file_type**: Format the data should be stored as. Currently the following formats are available: "rds" (default R-data format), "cs2" (cellular standard MAgPIE format), "cs2b" (cellular standard MAgPIE format with suppressed header ndata=1), "csv" (regional standard MAgPIE format), "cs3" (Format for multidimensional MAgPIE data, compatible to GAMS), "cs4" (alternative multidimensional format compatible to GAMS, in contrast to cs3 it can also handle sparse data), "csvr", "cs2r", "cs3r" and "cs4r" which are the same formats as the previous mentioned ones with the only difference that they have a REMIND compatible format, "m" (binary MAgPIE format "magpie"), "mz" (compressed binary MAgPIE format "magpie zipped"), "asc" (ASCII grid format), "nc" (netCDF format), "tif" (GEOtiff format) and "grd" (native raster format). If `file\_type=NULL` the file ending of the `file\_name` is used as format. If format is different to the formats mentioned standard MAgPIE format is assumed. Please be aware that the `file\_name` is independent of the `file\_type` you choose here, so no additional file ending will be added!
- **append**: Decides whether an existing file should be overwritten (FALSE) or the data should be added to it (TRUE). Append = TRUE only works if the existing data can be combined with the new data using the mbind function
- **comment**: Vector of strings: Optional comment giving additional information about the data. If different to NULL this will overwrite the content of attr(x,"comment")
- **comment.char**: character: a character vector of length one containing a single character or an empty string. Use ‘”’ to turn off the interpretation of comments altogether.
- **mode**: File permissions the file should be written with as 3-digit number (e.g. "777" means full access for user, group and all, "750" means full access for user, read access for group and no acces for anybody else). Set to NULL system defaults will be used. Access codes are identical to the codes used in unix function chmod.

... additional arguments passed to specific write functions

Details

This function can write 13 different MAgPIE file\_types. "cs2" is the new standard format for cellular data with or without header and the first columns (year,regiospatial) or only (regiospatial), "cs2b" is identical to "cs2" except that it will suppress the data name if it has only 1 element in the data dimension. "csv" is the standard format for regional data with or without header and the first columns (year,region,cellnumber) or only (region,cellnumber), "cs3" is another csv format which is specifically designed for multidimensional data for usage in GAMS. All these variants are written without further specification. "rds" is a R-default format for storing R objects. "magpie" (.m) and "magpie zipped" (.mz) are new formats developed to allow a less storage intensive management of MAgPIE-data. The only difference between both formats is that .mz is gzipped whereas .m is
not compressed. So .mz needs less memory, whereas .m might have a higher compatibility to other languages. "asc" is the ASCII grid format. "nc" is the netCDF format. It can only be applied for half degree data and writes one file per year per data column. In the case that more than one year and data column is supplied several files are written with the structure filename_year_datacolumn.asc

Note

The binary MAgPIE formats .m and .mz have the following content/structure (you only have to care for that if you want to implement read.magpie/write.magpie functions in other languages):

```
[ FileFormatVersion | Current file format version number (currently 6) | integer | 2 Byte ]
[ nchar_comment | Number of character bytes of the file comment | integer | 4 Byte ]
[ nbyte_metadata | Number of bytes of the serialized metadata (currently = 0) | integer | 4 Byte ]
[ nchar_sets | Number of characters bytes of all regionnames + 2 delimiter | integer | 4 Byte ]
[ nyears | Number of years | integer | 2 Byte ]
[ yearList | All years of the dataset (0, if year is not present) | integer | 2*nyears Byte ]
[ ncells | Number of cells | integer | 4 Byte ]
[ nchar_cell | Number of characters bytes of all regionnames + (nreg-1) for delimiters | integer | 4 Byte ]
[ cells | Cell names saved as cell1\cell2 (\n is the delimiter) | character | 1*nchar_cell Byte ]
[ nelem | Total number of data elements | integer | 4 Byte ]
[ nchar_data | Number of char. bytes of all datanames + (ndata - 1) for delimiters | integer | 4 Byte ]
[ datanames | Names saved in the format data1\ndata2 (\n as del.) | character | 1*nchar_data Byte ]
[ data | Data of the MAgPIE array in vectorized form | numeric | 4*nelem Byte ]
[ comment | Comment with additional information about the data | character | 1*nchar_comment Byte ]
[ sets | Set names with "\n as delimiter | character | 1*nchar_sets Byte ]
[ metadata | serialized metadata information (currently not in use) | bytes | 1*nbyte_metadata Byte ]
```

Author(s)

Jan Philipp Dietrich, Stephen Bi, Florian Humpenoeder

See Also

"magpie", read.magpie, mbind

Examples

```r
# a <- read.magpie("lpj_yield_ir.csv")
# write.magpie(a,"lpj_yield_ir.mz")
```
write.report

Write file in report format

Description

This function writes the content of a MAgPIE object into a file or returns it directly using the reporting format as it is used for many model inter-comparisons.

Usage

```r
write.report(
  x,  
  file = NULL,  
  model = NULL,  
  scenario = NULL,  
  unit = NULL,  
  ndigit = 4,  
  append = FALSE,  
  skipempty = TRUE,  
  extracols = NULL
)
```

Arguments

- **x**  
  MAgPIE object or a list of lists with MAgPIE objects as created by `read.report`. In the latter case settings for model and scenario are overwritten by the information given in the list.

- **file**  
  file name the object should be written to. If NULL the formatted content is returned

- **model**  
  Name of the model which calculated the results

- **scenario**  
  The scenario which was used to get that results.

- **unit**  
  Unit of the data. Only relevant if unit is not already supplied in Dimnames (format "name (unit)"). Can be either a single string or a vector of strings with a length equal to the number of different data elements in the MAgPIE object

- **ndigit**  
  Number of digits the output should have

- **append**  
  Logical which decides whether data should be added to an existing file or an existing file should be overwritten

- **skipempty**  
  Determines whether empty entries (all data NA) should be written to file or not.

- **extracols**  
  names of dimensions which should appear in the output as additional columns

Author(s)

Jan Philipp Dietrich
See Also

read.report

Examples

write.report(maxample("pop"))

write.report2  Write file in report format

Description

This function is deprecated, please use write.report instead.

Usage

write.report2(...)  

Arguments

... arguments are forwarded to write.report

Author(s)

Jan Philipp Dietrich

See Also

write.report
Index

* ~~
  * colSums-methods, 11
  * rowSums-methods, 56
* classes
  * magpie-class, 37
* keyword(s)
  * colSums-methods, 11
  * rowSums-methods, 56
* methods
  * as.array-methods, 6
  * as.data.frame-methods, 6
  * colSums-methods, 11
  * rowSums-methods, 56
  * setNames-methods, 58
* other
  * colSums-methods, 11
  * rowSums-methods, 56
* possible
  * colSums-methods, 11
  * rowSums-methods, 56
  * [,magpie,ANY,ANY-method (magpie-class), 37
  * [,magpie-method (magpie-class), 37
  * [<- ,magpie,ANY,ANY-method (magpie-class), 37
  * [<- ,magpie-method (magpie-class), 37
  * as.array,ANY-method (as.array-methods), 6
  * as.array-methods, 6
  * as.data.frame-method (as.data.frame-methods), 6
  * as.data.frame-methods, 6
  * as.data.frame-method (as.data.frame-methods), 6
  * as.data.frame-methods, 6
  * as.data.frame,ANY-method (as.data.frame-methods), 6
  * as.data.frame,ANY-method (as.data.frame-methods), 6
  * as.data.frame,magpie-method (as.data.frame-methods), 6
  * as.data.frame,magpie-method (as.data.frame-methods), 6
  * as.data.frame-methods, 6
  * as.magpie, 21, 41, 42, 49
  * as.magpie (magpie-class), 37
  * as.magpie, array-method (magpie-class), 37
  * as.magpie, data.frame-method (magpie-class), 37
  * as.magpie, logical-method (magpie-class), 37
  * as.magpie, lpj-method (magpie-class), 37
  * as.magpie, magpie-method (magpie-class), 37
  * as.magpie, NULL-method (magpie-class), 37
  * as.magpie, numeric-method (magpie-class), 37
  * as.magpie, quitte-method (magpie-class), 37
  * as.magpie, RasterBrick-method (magpie-class), 37
  * as.magpie, RasterLayer-method (magpie-class), 37
  * as.magpie, RasterStack-method (magpie-class), 37
  * as.magpie, tbl_df-method (magpie-class), 37
  * as.magpie-methods (magpie-class), 37
  * as.RasterBrick, 7, 24
  * clean_magpie, 8, 12
  * collapseDim, 9, 10
  * collapseNames, 9, 10, 46
  * colMeans, ANY-method (colSums-methods), 11
  * colMeans, magpie-method (colSums-methods), 11
  * colMeans-methods (colSums-methods), 11
  * colSums, ANY-method (colSums-methods), 11
  * colSums, magpie-method (colSums-methods), 11
  * colSums-methods, 11
  * complete_magpie, 11

67
INDEX

convergence, 12, 60
copy.attributes, 13
copy.attributes<- (copy.attributes), 13
copy.magpie, 14
dimCode, 4, 9, 15, 16, 19, 24–26, 57
dimExists, 16
dimOrder, 17
dimReduce, 18
dimSums, 19
escapeRegex, 20
fulldim, 20, 60, 62
getCells, 21
getCells<- (getCells), 21
getComment, 22
getComment<- (getComment), 22
getCoords, 7, 23, 32, 33
getCoords<- (getCoords), 23
getCPR, 22, 23, 24–30, 32, 38
getDim, 16
getItems, 9, 10, 19, 24, 26, 57
getItems<- (getItems), 26
getNames, 22, 23, 27, 29, 30, 32, 38, 58
getNames<- (getNames), 27
getRegionList, 28, 29
getRegionList<- (getRegionList), 28
getRegions, 22–24, 28, 29, 30, 32, 38
getRegions<- (getRegions), 29
getSets, 30, 49
getSets<- (getSets), 30
ggetYears, 23, 28–30, 31, 38
ggetYears<- (getYears), 31
grep, 20
hasCoords, 32
hasSets, 33
head, 34
head.magpie, 34
ifelse, magpie-method (magpie-class), 37
is.finite, magpie-method (magpie-class), 37
is.infinite, magpie-method (magpie-class), 37
is.magpie (magpie-class), 37
is.na, magpie-method (magpie-class), 37
is.nan, magpie-method (magpie-class), 37
is.spatial (is.temporal), 35
is.temporal, 35
isYear, 35
lowpass, 36
magclass (magclass-package), 4
magclass-package, 4
magclassdata, 37
magpie, 8–10, 22–24, 28–30, 32, 40, 44, 46, 53, 55, 64
magpie-class, 37
magpie_expand, 40, 41
magpie_expand_dim, 41
magpiesort, 39
magapply, 42
maxexample, 43
mbind, 5, 44, 64
mcalc, 25, 45
mcalc<- (mcalc), 45
mselect, 16, 45, 46
mselect<- (mselect), 46
ncells, 38, 47
ndata, 28, 38
ndata (ncells), 47
ndim, 48
new.magpie, 48
regions (ncells), 47
nyears, 38
nyears (ncells), 47
Ops, magpie, magpie-method (magpie-class), 37
Ops, magpie, numeric-method (magpie-class), 37
Ops, numeric, magpie-method (magpie-class), 37
options, 41, 42
population_magpie, 50
print, 51
print.magpie, 50
read.magpie, 15, 22–24, 28–30, 32, 38, 51, 64
read.report, 53, 66
replace_non_finite, 55
round, magpie-method (round-methods), 55
round-methods, 55
rowMeans, ANY-method (rowSums-methods), 56
rowMeans, magpie-method (rowSums-methods), 56
rowMeans-methods (rowSums-methods), 56
rowSums, 19
rowSums, ANY-method (rowSums-methods), 56
rowSums, magpie-method (rowSums-methods), 56
rowSums-methods, 56

setCells (getCells), 21
setComment (getComment), 22
setItems, 57
setNames, 22, 32
setNames (setNames-methods), 58
setNames, magpie-method (setNames-methods), 58
setNames, NULL-method (setNames-methods), 58
setNames-methods, 58
setYears (getYears), 31
sizeCheck, 58

tail, 34
tail.magpie (head.magpie), 34
time_interpolate, 59

unwrap, 21, 60, 62

where, 61
wrap, 21, 60, 61
write.magpie, 15, 22–24, 28–30, 32, 38, 53, 62
write.report, 54, 65, 66
write.report2, 66