

Package: `piamModelTests` (via `r-universe`)

February 4, 2025

Type Package

Title `piamModelTests` Tools

Version 0.31.5

Date 2023-12-06

Description A collection of R tools provided by the Integrated Assessment Modeling Consortium (IAMC) for data analysis and diagnostics.

License MIT + file LICENSE

Depends R(>= 2.10.0)

Imports `quitte`(>= 0.3066), `magclass`(>= 4.40), `mip`(>= 0.97), `lusweave`, `dplyr`, `utils`, `methods`, `reshape2`, `readxl`, `writexl`, `tibble`

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Suggests `covr`, `knitr`, `rmarkdown`

VignetteBuilder `knitr`

Config/pak/sysreqs `make libicu-dev libssl-dev libx11-dev zlib1g-dev`

Repository <https://pik-piam.r-universe.dev>

RemoteUrl <https://github.com/pik-piam/piamModelTests>

RemoteRef HEAD

RemoteSha 794a95147a6e15da2256df94c5d9ee0c9c7afb49

Contents

<code>piamModelTests-package</code>	2
<code>codedAsInteger</code>	2
<code>collectFunctions</code>	3
<code>createInputData</code>	4
<code>example_REMIND</code>	4
<code>filterInputData</code>	5

getShiftFactor	5
iamCheck	6
iamProjectConfig	7
iamReferenceData	8
iamSummaryPDF	8
preCheckDuplicates	9
processCheck	10
RenameAndAggregate	11
write.reportProject	12

Index	14
--------------	-----------

piamModelTests-package

piamModelTests: piamModelTests Tools

Description

A collection of R tools provided by the Integrated Assessment Modeling Consortium (IAMC) for data analysis and diagnostics.

Author(s)

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

Authors:

- Cornelia Auer
- Anastasis Giannousakis
- Christoph Bertram
- Falk Benke
- Florian Humpenoeder
- Lavinia Baumstark

codedAsInteger

codedAsInteger

Description

Encodes data as integer values(corresponding to their factor representatives). NA values are exchanged by 0.

Usage

`codedAsInteger(input)`

Arguments

input Input data that should be checked, provided as a object related to a data frame.

Value

values encoded as integers without NAs.

collectFunctions *collectFunctions*

Description

Collects functions which follow a given name pattern and compares their arguments against a list of allowed arguments

Usage

```
collectFunctions(  
  pattern = "^check",  
  globalenv = FALSE,  
  allowed_args = c("x", "cfg")  
)
```

Arguments

pattern Name pattern the function name should match. Default is to collect functions starting with "check"

globalenv Boolean deciding whether functions in the global environment should be considered or not.

allowed_args Vector of allowed arguments. If a function contains an argument not listed here it will be ignored and a warning will be returned

Value

a character vector of function calls fulfilling all requirements

Author(s)

Jan Philipp Dietrich

See Also

[iamCheck](#)

Examples

```
collectFunctions()
```

createInputData	<i>createInputData</i>
-----------------	------------------------

Description

Takes given data and provides it in the required data format (magpie)

Usage

```
createInputData(x, cfg = "CDLINKS", ref = "IAMC", verbose = TRUE, ...)
```

Arguments

x	Input data that should be checked, provided as a file path to a reporting file, a quitte object or an object which can be converted to quitte using as.quitte
cfg	Project configuration that should be used (currently available: "CDLINKS"). Either a project name, a path to a config file or a data frame specifying available variables and corresponding properties as returned by iamProjectConfig() .
ref	Reference data for comparison. Either a project name (currently available: "IAMC"), a path to a mif file or a quitte object containing the data.
verbose	Boolean influencing the degree of information returned by the function. verbose=TRUE returns detailed information whereas verbose=FALSE returns a summary.
...	additional data objects which are forwarded to the check functions

Value

Input named list with elements available for check functions

Author(s)

Cornelia Auer

example_REMIND	<i>example_REMIND</i>
----------------	-----------------------

Description

Example output from the REMIND model

Value

data generated from the REMIND model

Author(s)

Cornelia Auer

filterInputData	<i>filterInputData</i>
-----------------	------------------------

Description

Filters given data (input) according to available default variables (cfg). Pre-checks are performed about the consistency of the input data (e.g. illegal or missing variables) and the status (out) returned.

Usage

```
filterInputData(input, cfg = "CDLINKS", globalenv = FALSE, out = NULL)
```

Arguments

input	Named list with elements available for check functions
cfg	Project configuration that should be used. Either a project name (currently available: "CDLINKS"), a path to a config file or a data frame specifying available variables and corresponding properties as returned by iamProjectConfig() .
globalenv	Boolean deciding whether functions in the global environment should be considered
out	List with status from pre-checks, e.g. illegal or missing variables.

Value

List with 1) filtered input and 2) status-output about the consistency of the input data

Author(s)

Cornelia Auer

getShiftFactor	<i>getShiftFactor</i>
----------------	-----------------------

Description

Method: Essentially computes order by the power of ten from a given positive number. The order value is then used to compute a shift factor, which is needed to uniquely represent the data values as integer in `preCheckDuplicates`.

Usage

```
getShiftFactor(positiveNumber)
```

Arguments

positiveNumber number ≥ 0

Value

order of power of ten for the input value

iamCheck	<i>iamCheck</i>
----------	-----------------

Description

Runs various diagnostics over a provided data set checking whether the provided data is in line with IAMC database guidelines.

Usage

```
iamCheck(
  x,
  pdf = NULL,
  cfg = "CDLINKS",
  refData = "IAMC",
  verbose = FALSE,
  globalenv = FALSE,
  pdfStyle = NULL,
  ...
)
```

Arguments

x	Input data that should be checked, provided as a file path to a reporting file, a quitte object or an object which can be converted to quitte using as.quitte
pdf	File name used for a PDF containing diagnostic results of the check. If set to NULL no pdf will be written.
cfg	Project configuration that should be used. Either a project name (currently available: "CDLINKS"), a path to a config file or a data frame specifying available variables and corresponding properties as returned by iamProjectConfig() .
refData	Reference data for comparison. Either a project name (currently available: "IAMC"), a path to a mif file or a quitte object containing the data.
verbose	Boolean influencing the degree of information returned by the function. verbose=TRUE returns detailed information whereas verbose=FALSE returns a summary.
globalenv	Boolean deciding whether functions in the global environment should be considered or not.
pdfStyle	list of style-options for the pdf
...	additional data objects which are forwarded to the check functions

Value

List of all inputs and outputs created by the performed checks (invisible)

Author(s)

Jan Philipp Dietrich

See Also

[iamProjectConfig](#), [as.quitte](#), [is.quitte](#)

Examples

```
# run check with example data
iamCheck(example_REMIND, cfg="CDLINKS")
```

<code>iamProjectConfig</code>	<code>iamProjectConfig</code>
-------------------------------	-------------------------------

Description

Function to return available variables and corresponding properties for a given project configuration.

Usage

```
iamProjectConfig(cfg = "CDLINKS")
```

Arguments

<code>cfg</code>	Project configuration that should be used. Either a project name (currently available: "CDLINKS"), a path to a config file or a data frame specifying available variables and corresponding properties
------------------	--

Value

Data frame containing available variables and corresponding properties

Author(s)

Jan Philipp Dietrich

See Also

[as.quitte](#), [is.quitte](#)

Examples

```
iamProjectConfig()
```

iamReferenceData *iamReferenceData*

Description

Function to return historical reference data for validation.

Usage

```
iamReferenceData(ref = "IAMC")
```

Arguments

`ref` Reference data for comparison. Either a project name (currently available: "IAMC"), a path to a file or a data frame containing the data.

Value

Quitte object containing the reference data set

Author(s)

Jan Philipp Dietrich

See Also

[as.quitte](#), [is.quitte](#)

Examples

```
iamReferenceData()
```

iamSummaryPDF *iamSummaryPDF*

Description

Creates a PDF summarizing check results and adding additional results for a comparison to reference data (e.g. matching to historical data)

Usage

```
iamSummaryPDF(
  input,
  check_results = NULL,
  file = "summary.pdf",
  maxLinesOutput = 200,
  pdfStyle = NULL,
  ...
)
```

Arguments

input	named list with elements available for check functions
check_results	list with check results as returned by iamCheck
file	File name the summary should be written to or a Sweave object. If a sweave object is provided the function will return the updated object, otherwise it will write its content to the file.
maxLinesOutput	maximum number of lines that should be output in the pdf
pdfStyle	list of style-options for the pdf
...	additional arguments sent to swclose

Author(s)

Jan Philipp Dietrich

See Also

[iamCheck](#), [iamProjectConfig](#)

Examples

```
## Not run:
input <- list(x=example_REMIND, ref=iamReferenceData())
check_results <- iamCheck(example_REMIND)

iamSummaryPDF(check_results$input, check_results$out)

## End(Not run)
```

```
preCheckDuplicates    preCheckDuplicates
```

Description

Checks for duplicate entries in the data. To optimize performance the single data values are encoded into integer values (corresponding to their factor representatives). The encoded integer values are then compared if duplicates exist.

Usage

```
preCheckDuplicates(x)
```

Arguments

x Input data that should be checked, provided as a quitte object [as.quitte](#)

Value

Two-dimensional list. 1) message with how many dpuplicates 2) list of duplicates (in their integer representation)

Author(s)

Cornelia Auer

processCheck	<i>processCheck</i>
--------------	---------------------

Description

Processes a check, which means that it returns a note if the check fails, returns a check summary based on the chosen verbosity and returns the check results in a structured way. All checks should be run within processCheck. Requirement for a check function is that it returns a list with 2 elements: "message" which contains the standard message that should show up for the test and "failed" which is a vector of names for which the corresponding test failed.

Usage

```
processCheck(check, input)
```

Arguments

check Function call as character that should be run
input Named list with elements available for check functions

Value

List containing check results

Author(s)

Jan Philipp Dietrich

See Also

[iamProjectConfig](#), [as.quitte](#), [is.quitte](#)

Examples

```
out <- processCheck(check = "checkMin(x, cfg)",
  input = list(x=example_REMIND, cfg=iamProjectConfig()))
```

RenameAndAggregate *RenameAndAggregate*

Description

Rename and aggregate data using a mapping

Reads in a substitutes names of variables according to the mapping, multiplies reported values by an optional factor in a column named "factor" of the mapping, and saves the output in a new *.mif

Usage

```
RenameAndAggregate(data, mapping, missing_log = NULL)
```

Arguments

data	Lists with list of magpie-objects (a magpie-object as created by read.report), first list contains scenarios, second list the models
mapping	mapping of the variable names of the read-in mif. The header is used for naming. The format of the mapping should be: 1st column the standard naming in PIK mif format. X further columns that contain the indicator names in the reporting format. Can also contain several indicator columns (e.g Variable and Item). Optional columns with reserved names are unit, weight, factor, and spatial. Factor is a number that the results will be multiplied with (e.g. to transform CO2 into C). Weight is needed if several mif indicators shall be aggregated to one reporting indicator. You always need a weight column if you have multiple mif to one reporting mappings. If you have a weight column, you have to have values in it for all indicators. If NULL, the results are added up; if you provide an indicator name (of a mif indicator), this indicator will be used for the weighting of a weighted mean. Spatial should be set to "glo" for mif indicators that shall only be reported globally and "reg" for only reporting locally. The default is "reg+glo", which implies reporting on global and local level. In the case of aggregation, contradicting entries in spatial column for the same reporting indicator will throw an error. Unit is a name of the unit without () Example: "magpie";"agmip";"item";"unit";"weight";"factor" "NutritionI+Calorie Supply (kcal/capita/day)";"CALO";"AGR";"kcal/capita/day";"NULL";1
missing_log	name of logfile to record variables which are present in the mapping but missing in the mif file. By default, no logfile is produced

Author(s)

Christoph Bertram, Lavinia Baumstark, Anastasis Giannousakis, Florian Humpenoeder, Falk Benke, Benjamin Leon Bodirsky

See Also[write.report](#)**Examples**

```
## Not run:
RenameAndAggregate(list(model=list(scenario=population_magpie)), "Mapping_generic_ADVANCE.csv")

## End(Not run)
```

```
write.reportProject Write file in specific project format
```

Description

Reads in a reporting.mif or uses a magpie object based on a read-in reporting.mif, substitutes names of variables according to the mapping, multiplies reported values by an optional factor in a column named "factor" of the mapping, and saves the output in a new *.mif

Usage

```
write.reportProject(
  mif,
  mapping,
  file = NULL,
  max_file_size = NULL,
  format = "default",
  append = FALSE,
  missing_log = NULL,
  ...
)
```

Arguments

mif	Lists with magpie-objects or a magpie-object as created by read.report or a path to a report.mif
mapping	mapping of the variable names of the read-in mif. The header is used for naming. The format of the mapping should be: 1st column the standard naming in PIK mif format. X further columns that contain the indicator names in the reporting format. Can also contain several indicator columns (e.g Variable and Item). Optional columns with reserved names are unit, weight, factor, and spatial. Factor is a number that the results will be multiplied with (e.g. to transform CO2 into C). Weight is needed if several mif indicators shall be aggregated to one reporting indicator. You always need a weight column if you have multiple mif to one reporting mappings. If you have a weight column, you have to have values in it for all indicators. If NULL, the results are added up; if you

provide an indicator name (of a mif indicator), this indicator will be used for the weighting of a weighted mean. Spatial should be set to "glo" for mif indicators that shall only be reported globally and "reg" for only reporting locally. The default is "reg+glo", which implies reporting on global and local level. In the case of aggregation, contradicting entries in spatial column for the same reporting indicator will throw an error. Unit is a name of the unit without () Example: "magpie";"agmip";"item";"unit";"weight";"factor" "Nutrition|+|Calorie Supply (kcal/capita/day)";"CALO";"AGR";"kcal/capita/day";"NULL";1

file	name of the output file, default=NULL returns the output object
max_file_size	maximum file size in MB; if size of file exceeds max_file_size reporting is split into multiple files
format	available reporting formats: "default", "IAMC" and "AgMIP". "default" and "IAMC" are very similar (wide format for year) and differ only in the use of semi-colon (default) and comma (IAMC) as separator. "AgMIP" is in long format.
append	Logical which decides whether data should be added to an existing file or an existing file should be overwritten
missing_log	name of logfile to record variables which are present in the mapping but missing in the mif file. By default, no logfile is produced
...	arguments passed to write.report and write.report2

Author(s)

Christoph Bertram, Lavinia Baumstark, Anastasis Giannousakis, Florian Humpenoeder, Falk Benke, Benjamin Leon Bodirsky

See Also

[write.report](#), [RenameAndAggregate](#)

Examples

```
## Not run:
write.reportProject("REMIND_generic_test.mif", "Mapping_generic_ADVANCE.csv")

## End(Not run)
```

Index

`as.quitte`, [4](#), [6–8](#), [10](#)

`codedAsInteger`, [2](#)

`collectFunctions`, [3](#)

`createInputData`, [4](#)

`example_REMIND`, [4](#)

`filterInputData`, [5](#)

`getShiftFactor`, [5](#)

`iamCheck`, [3](#), [6](#), [9](#)

`iamProjectConfig`, [4–7](#), [7](#), [9](#), [10](#)

`iamReferenceData`, [8](#)

`iamSummaryPDF`, [8](#)

`is.quitte`, [7](#), [8](#), [10](#)

`piamModelTests`

 (`piamModelTests-package`), [2](#)

`piamModelTests-package`, [2](#)

`preCheckDuplicates`, [9](#)

`processCheck`, [10](#)

`RenameAndAggregate`, [11](#), [13](#)

`swclose`, [9](#)

`write.report`, [12](#), [13](#)

`write.reportProject`, [12](#)