Package: mip (via r-universe)

September 6, 2024

Type Package

Title Comparison of multi-model runs

Version 0.149.3

Date 2024-09-06

Description Package contains generic functions to produce comparison plots of multi-model runs.

License BSD_2_clause + file LICENSE

URL https://github.com/pik-piam/mip, https://doi.org/10.5281/zenodo.1158586

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

Depends R (>= 2.10.0), magclass, quitte (>= 0.3072)

Imports data.table, dplyr, ggplot2, gridExtra, htmltools, lusweave (>= 1.43.2), plotly, RColorBrewer, reshape2, rlang, shiny, stringr, tidyr, trafficlight, withr,

Suggests gdxrrw, knitr, rmarkdown, testthat

VignetteBuilder knitr

Encoding UTF-8

LazyData yes

RoxygenNote 7.3.2

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

RemoteUrl https://github.com/pik-piam/mip

RemoteRef HEAD

RemoteSha 158a58f8cfb9a21b0d7e9fd3273108b89f45d0d3

Contents

mip-package 2
calculateRatio 3
checkGlobalOptionsProvided 4

dataframeFromGdx . . . 4
extractVariableGroups . . . 5
getLegend . . . 6
getPlotData . . . 6
harmonize . . . 7
identifierModelScen . . . 8
longestCommonPrefix . . . 8
mipArea . . . 9
mipBarYearData . . . 10
mipIterations . . . 11
mipLineHistorical . . . 12
mip_example_data . . . 14
plotPercentiles . . . 15
plotstyle . . . 16
plotstyle.add . . . 17
scenTool . . . 18
scenToolMAgPIE . . . 19
scratchBar . . . 20
shorten_legend . . . 20
showAreaAndBarPlots . . . 21
showAreaAndBarPlotsPlus . . . 23
showLinePlots . . . 24
showLinePlotsWithTarget . . . 25
showMultiLinePlots . . . 26
showMultiLinePlotsByVariable . . . 27
showRegiLinePlots . . . 29
sideBySidePlots . . . 30
theme_mip . . . 31
validationpdf . . . 31
warnMissingVars . . . 32

Index **34**

`mip-package` *The MIP R package*

Description

Contains the routines for plotting multi model and multi scenario comparisons

Details

Package:	mip
Type:	Package
Version:	7.6
Date:	2016-06-13
License:	LGPL-3

LazyLoad: yes

Author(s)

David Klein

Maintainer: Anastasis Giannousakis <giannou@pik-potsdam.de>

See Also

Useful links:

- https://github.com/pik-piam/mip
- doi:10.5281/zenodo.1158586
- Report bugs at https://github.com/pik-piam/mip/issues

`calculateRatio` *Calculate Ratios for Quitte Objects.*

Description

Changes the value of variables given in `numerators` by dividing by `denominator` and multiplying `conversionFactor`. Sets `unit` of these variables to `newUnit`.

Usage

```
calculateRatio(
  data,
  numerators,
  denominator,
  newUnit = "1",
  conversionFactor = 1
)
```

Arguments

`data`	A quitte object.
`numerators`	A character vector. Entries in the variable column of data.
`denominator`	A single string. An entry in the variable column of data.
`newUnit`	A single string.
`conversionFactor`	A single numerical value.

Value

A quitte object with changed values and new unit. Unused levels are dropped.

checkGlobalOptionsProvided *Check Whether Global Options Are Provided*

Description

The function checks whether a variable of the name `optNames` is available. If not a warning message is produced, indicating how this variable can be set using `options()`.

Usage

```
checkGlobalOptionsProvided(optNames, envir = rlang::caller_env())
```

Arguments

`optNames`	A character vector. The names of the variables.
`envir`	The environment where to look for the variables.

Value

Returns `NULL` invisibly.

dataframeFromGdx *dataframeFromGdx*

Description

Read data for a given symbol from a gdx file and return it as a dataframe.

Usage

```
dataframeFromGdx(symbolName, pathToGdx, ...)
```

Arguments

`symbolName`	The name of a symbol to be extracted from gdx.
`pathToGdx`	Path to a gdx file.
`...`	Additional arguments passed to gdxrrw::rgdx.

Details

This function is similar to gdxrrw::rgdx.param, but it also works for variables.

Value

A data frame with data about symbolName from the given gdx file. All columns are character vectors, except the last column which holds numeric values, because that is the format that rgdx returns.

Author(s)

Pascal Führlich

See Also

getPlotData

`extractVariableGroups` *extractVariableGroups*

Description

Groups variable names by summation groups based on the |+| separators given in the variable names. If no |+| are present the function will try to derive summation groups based on | separators.

Usage

```
extractVariableGroups(x, keepOrigNames = FALSE, sorted = FALSE)
```

Arguments

`x`	a vector of variable names
`keepOrigNames`	if set, the returned list contains the original variables (to the value of which the grouped ones have to sum up) as names instead of made up group names, if they exist.
`sorted`	boolean, indicating whether the variables within each group should be returned alp

Value

a named list of variable groups with group name as name and vector of entities as content

Author(s)

Anastasis Giannousakis, David Klein, Jan Philipp Dietrich

See Also

plotstyle.add

Examples

```
x <- c("a|+|1|+|aa","a|+|2|abc","a|+|1|+|bb","a|+|1|+|cc","a|+|3|+|aa","a|+|3|+|bb")
mip::extractVariableGroups(x)
```

---

getLegend *Get Legend of a ggplot.*

Description

The function extracts the legend of a ggplot object.

Usage

```
getLegend(plt)
```

Arguments

plt	A ggplot object.

Value

The legend of plt or NULL if no legend was found.

---

getPlotData *getPlotData*

Description

Get ready-to-plot data from one or more gdx files.

Usage

```
getPlotData(symbolName, pathToGdx = ".", ...)
```

Arguments

symbolName	The name of a symbol to be extracted from gdx.
pathToGdx	Path to one or more gdx files or a a path to a folder with fulldata gdx files. If multiple paths are provided each one represents data after a specific iteration. The order of paths should match iteration order, e.g. pathToGdx[1] should hold data for the first iteration, pathToGdx[2] for the second iteration etc. If the path to a folder is given the fulldata gdx files in it are used.
...	Additional arguments passed to gdxrrw::rgdx.

Value

A data frame with data from the given gdx file(s). If multiple gdx files are provided an additional "iteration" column is added. The iteration value will be 1 for data rows from the first gdx, 2 for the second etc. The last column will always be the actual value column called <symbolName>.

Author(s)

Pascal Führlich

See Also

mipIterations, dataframeFromGdx

`harmonize` *Harmonization of model data to historical data, using harmonization methods of aneris, ported to R. See: https://github.com/iiasa/aneris/blob/ad6301eb42155c968f20b2c7e071cbec039acc03/aneris/meth*

Description

Harmonization of model data to historical data, using harmonization methods of aneris, ported to R.
See: https://github.com/iiasa/aneris/blob/ad6301eb42155c968f20b2c7e071cbec039acc03/aneris/methods.py

Usage

```
harmonize(
  df,
  hist,
  finalYear = "2050",
  harmonizeYear = "2015",
  method = "ratio",
  suffix = ""
)
```

Arguments

`df`	data frame with model data to be harmonized, must have the following columns: variable, region, scenario, model, period
`hist`	data frame with historical data to be used for harmonization, must also have the following columns: variable, region, scenario, model, period
`finalYear`	when should harmonized data match model data again?
`harmonizeYear`	when should harmonization begin? sets model data = reference data for this year
`method`	harmonization method, currently supported methods are "ratio" and "offset"
`suffix`	to be appended to harmonized variables

Author(s)

Falk Benke

identifierModelScen *add identifier based on model and scenario names*

Description

add identifier based on model and scenario names

Usage

```
identifierModelScen(x)
```

Arguments

x	A quitte object

Value

A factor. If more than one model but only one scenario, use model. If more than one scenario but only one model, use scenario. Else, combine them.

longestCommonPrefix *Longest Common Prefix*

Description

Longest Common Prefix

Usage

```
longestCommonPrefix(x)
```

Arguments

x	A character vector.

Value

A single string. The longest common prefix of x.

mipArea *Generic area plot function. Automatically creates facet grid from data. Optionally adds total line.*

Description

Generic area plot function. Automatically creates facet grid from data. Optionally adds total line.

Usage

```
mipArea(
  x,
  stack_priority = c("variable", "region"),
  total = TRUE,
  scales = "fixed",
  shorten = TRUE,
  hist = NULL,
  hist_source = "first",
  ylab = NULL
)
```

Arguments

x	Data to plot. Allowed data formats: magpie or quitte. NOTE: To ensure correct conversion to quitte objects, the dimension that contains the variables must have one of the following names: variable, scenario, or model.
stack_priority	Name of column you want to stack. If you provide more than one column name the function will scan the columns in the given order and use the first dimension for stacking that has more than one element.
total	total data to plot. Allowed inputs: magpie, quitte or boolean. If total data is provided by user in magpie or quitte format it will be added to the plot. If user sets total to TRUE total will be calculated by the function and added to the plot. If total is FALSE the plot will ignore it.
scales	scales can be fixed ("fixed", default), free ("free"), or free in one dimension ("free_x", "free_y")?
shorten	Shorten variable names (default is TRUE) by removing categories only if they are identical (for short names in the legend)
hist	Historical data. Allowed data formats: magpie or quitte. NOTE: To ensure correct conversion to quitte objects, the dimension that contains the variables must have one of the following names: variable, scenario, model.
hist_source	If there are multiple historical sources the name of the source that you want to be plotted.
ylab	y-axis text

Example Plot

Author(s)

David Klein, Jan Philipp Dietrich

Examples

```
p <- mipArea(x = mip_example_data)
# create plot with best-guess design (internally using theme_mip(size=12))
p <- mipArea(mip_example_data)
# override default theme with theme_grey and move legend to top
library(ggplot2)
p <- p + theme_grey() + theme(legend.position = "top")
# go back to theme_mip and increase font size
p <- p + theme_mip(size = 18)
# change facetting
p <- p + facet_grid(region ~ scenario)
```

---

mipBarYearData *mipBarYearData*

Description

Function for plotting (bar-plot) MAgPIE objects and compare different scenarios or models, on the x-axis for some time steps one bar for each scenario/model is generated

Usage

```
mipBarYearData(
  x,
  colour = NULL,
  ylab = NULL,
  xlab = NULL,
  title = NULL,
  scenario_markers = TRUE
)
```

Arguments

x	Data to plot. Allowed data formats: magpie or quitte
colour	Dimension to be colored, default: "Scenario"
ylab	y-axis text
xlab	x-axis text
title	title appearing at the top of the plot

scenario_markers
: Use markers to conserve space with long scenario names. Symbols are either picked automatically (default), or can be passed as a named vector in the form of `c('scenario' = 'marker')`, where marker is a number between 1 and 20, or a ggplot2 shape name (see `vignette("ggplot2-specs")`). Set to `FALSE` to not use markers.

Example Plot

Author(s)

Lavinia Baumstark, Oliver Richters

Examples

```
## Not run:
plotCompBarYearData(EnInv, ylab = "Energy Investments|Elec (billion US$2005/yr)",
                    colour = plotstyle(getNames(EnInv, dim = 2)))

## End(Not run)
```

---

mipIterations *plotIterations*

Description

Creates interactive line plots using ggplot and plotly. Creates one plot for each combination of values in columns not plotted via this function's arguments. If the special value "year" is passed as xAxis, color, slider or facets a list of possible column names representing years (e.g. "ttot", "tall", "t_all") is checked, the first one in names(x) is used.

Usage

```
mipIterations(
  plotData,
  returnGgplots = FALSE,
  xAxis = "year",
  color = NULL,
  slider = "iteration",
  facets = "region",
  facetScales = "fixed"
)
```

Arguments

plotData	A data frame. The actual value column must be the last. Use mip::getPlotData to get a ready-to-plot data frame from one or more gdx files.
returnGgplots	If FALSE (the default) show interactive plotly plots with slider support. Set to TRUE to return ggplots which can be customized, but are not interactive. To re-enable slider support and interactivity run lapply(ggplots, plotly::ggplotly) after customizing the ggplots.
xAxis	A string from names(x), defining which column is plotted on the x-axis of the plots. Must not be NULL.
color	A string from names(x), defining which column is plotted as color. If NULL color is not used.
slider	A string from names(x), defining which column is plotted as a slider. The slider requires plotly. If NULL no slider is used.
facets	A string from names(x), defining which column is used for grouping. A small plot (facet) is shown for each group. If NULL facets are not used.
facetScales	The 'scales' argument for facets (if used), defaults to 'fixed'. See help(facet_wrap) for more info.

Value

A list of plotly plots, if returnGgplots is TRUE a list of ggplots instead

Author(s)

Pascal Führlich

See Also

getPlotData

---

mipLineHistorical *Compares data by producing line plot*

Description

Compares data by producing line plot

Usage

```
mipLineHistorical(
  x,
  x_hist = NULL,
  color.dim = "identifier",
  linetype.dim = NULL,
  facet.dim = "region",
```

```
  funnel.dim = NULL,
  ylab = NULL,
  xlab = "Year",
  title = NULL,
  color.dim.name = NULL,
  ybreaks = NULL,
  ylim = 0,
  ylog = NULL,
  size = 14,
  scales = "fixed",
  leg.proj = FALSE,
  plot.priority = c("x", "x_hist", "x_proj"),
  ggobject = TRUE,
  paper_style = FALSE,
  xlim = NULL,
  facet.ncol = 3,
  legend.ncol = 1,
  hlines = NULL,
  hlines.labels = NULL,
  color.dim.manual = NULL,
  color.dim.manual.hist = NULL
)
```

Arguments

x	Data to plot. Allowed data formats: magpie or quitte
x_hist	historical data to plot. Allowed data formats: magpie or quitte, If no historic information is provided the plot will ignore it.
color.dim	dimension used for different colors, default="identifier"; can only be chosen freely if x_hist is NULL.
linetype.dim	dimension used for different line types, default=NULL
facet.dim	dimension used for the facets, default="region"
funnel.dim	dimension used for different funnels, default=NULL
ylab	y-axis label
xlab	x-axis label, default="Year"
title	title of the plot
color.dim.name	name for the color-dimension used in the legend
ybreaks	add breaks for the y axis
ylim	y limits
ylog	=T if the-axis should be logarithmic
size	text size in the plot
scales	Are scales shared across all facets (the default, "fixed"), or do they vary across rows ("free_x"), columns ("free_y"), or both rows and columns ("free")
leg.proj	to add a detailed legend for the projected data. Default is FALSE.

plot.priority	Sets the order of ploting and overlap of the data by specifying a vector of three stirng elements. Argument x stands for model output, x_hist is for obeserved (historical data) and x_proj is for projected data from other models.
ggobject	returns a ggplot object. Default is TRUE.
paper_style	removes grey color from facets if TRUE Default is FALSE.
xlim	x axis limits as vector with min and max year
facet.ncol	number of columns used for faceting, default=3.
legend.ncol	number of columns used in legends, default=1.
hlines	optional horizontal lines to be added to the plot, Allowed data formats: magpie, Default is NULL.
hlines.labels	optional labels for horizontal lines, Allowed data formats: named vector, where each name corresponds to exactly one variable in hlines, Default is NULL.
color.dim.manual	optional vector with manual colors replacing default colors of color.dim, default is NULL.
color.dim.manual.hist	optional vector with manual colors replacing default colors of color.dim for historical data, default is NULL.

Example Plot

Author(s)

Lavinia Baumstark, Mishko Stevanovic, Florian Humpenoeder

Examples

```
  ## Not run:
    p <- mipLineHistorical(x,x_hist=hist,ylab="example",xlab="Year",title=NULL)

## End(Not run)
```

---

mip_example_data *mip_example_data*

Description

mip_example_data

Value

Regional sulfur emissions from CRESCENDO project

Author(s)

David Klein

plotPercentiles *Comparison line plots with percentiles*

Description

Line plots show median (50th percentile) of user selected variable(s) obtained from different scenario runs. If available in the data, ribbon plots will also show the 33th - 67th percentile region in a darker color and the 5th – 95th percentile region in a lighter color. Note: the 5th, 33th, 67th and 95th percentiles must be provided in the data set as the percentiles are not computed

Usage

```
plotPercentiles(df, scenarios = NULL, variables = NULL)
```

Arguments

df	The quitte-style data frame must contain all percentiles of the quantity of interest as individual variables (e.g. for atmospheric CO2 concentrations "Atmospheric Concentrations|CO2|50th Percentile", "Atmospheric Concentrations|CO2|33th Percentile", ..., must be present)
scenarios	Character vector containing names of the desired scenarios. If NULL, all scenarios present in the data will be displayed
variables	Character vector containing names of the desired variables. If NULL, all variables present in the data will be displayed. When selecting particular variables for display only use the "Any|Variable"-prefix and omit the "X-th Percentile"-suffix (e.g. for atmospheric CO2 concentrations write "Atmospheric Concentrations|CO2")

Example Plot

Author(s)

Tonn Rüter

Examples

```
## Not run:
# Plot atmospheric CO2 concentrations for all scenarios available in the data
p <- plotPercentiles(
  data,
  # Use variable name without "X-th Percentile"-suffix
  variables = c("AR6 climate diagnostics|Atmospheric Concentrations|CO2|MAGICCv7.5.3")
)
# Plot all available variables for selected scenarios
p <- plotPercentiles(data, scenarios = c("d_delfrag", "d_another"))

## End(Not run)
```

`plotstyle` *Returns plot styles for given entities*

Description

Returns a named vector (using entity names) with style codes (e.g. colors) for given entities.

Usage

```
plotstyle(
  ...,
  out = "color",
  unknown = NULL,
  plot = FALSE,
  verbosity = getOption("plotstyle.verbosity"),
  regexp = FALSE,
  strip_units = getOption("plotstyle.strip_units", default = TRUE)
)
```

Arguments

`...`	One or more strings or a vector of strings with names of entities (regions, variable names, etc.). Units in brackets "(US$2005/GJ)" will be ignored. If left empty all available entities will be used
`out`	Switch defining which type of plot style you want to get: Currently you can choose between "color", "legend" and "all" (the latter will return a dataframe with all available plot styles)
`unknown`	Optional data frame defining plot styles for unknown entities. A default color map will be used for unknown entities if nothing is provided here
`plot`	If TRUE plots with all given entities and their colors will be produced (to produce plots with all available entities leave the `...` entry empty!)
`verbosity`	Set to 1 if you want to know for which unknown entities plotstyle brewed colors
`regexp`	If `TRUE`, match entities as regular expressions. Matching entities are expanded, non-matching entities are returned as the original expression. Does not generate default color maps. Implies `plot = FALSE` and `verbosity = 0`.
`strip_units`	If `TRUE` everything from the first opening brace (`'('`) on is stripped from the entity names. Defaults to `TRUE` and can be set globally using the `plotstyle.strip_units` option.

Value

Plot styles for given entities

Colors for unknown entities

Author(s)

David Klein, Jan Philipp Dietrich

See Also

plotstyle.add

Examples

```
entities <- c("AFR", "AAA", "AFR", "UNKNOWN_ELEMENT2")
plotstyle(entities)
unknown <- data.frame(row.names = c("AAA", "UNKNOWN_ELEMENT2"),
                  color = c("#123456", "#345678"),
                  legend = c("l_AAA", "l_UNKNOWN_ELEMENT2"))
plotstyle(entities, unknown = unknown)
plotstyle(entities, out = "legend")
plotstyle(entities, out = "all")

# search for all 'Final Energy Biomass' entities
plotstyle("^Final Energy\\|.*Biomass", regexp = TRUE)

# search for all three-letter entities (a.k.a. regions)
plotstyle("^[A-Z]{3}$", regexp = TRUE, out = "all")
```

---

plotstyle.add *Adds plot styles locally to plotstyle.csv*

Description

Adds plot styles locally to plotstyle object and returns a dataframe with all plotstyles including the added data. However, it does NOT change the "./inst/extdata/plotstyle.csv". To add new entities to "./inst/extdata/plotstyle.csv" please open the file in your editor and add or change values by hand. By default plotstyles of already existing entities will not be changed. Only new entities will be added. Use the `replace` switch to replace existing styles. If you want to keep the legend text or the color of an already existing entity and only replace one of the two values, use the string "keep" for the value you want to keep.

Usage

```
plotstyle.add(
  entity,
  legend,
  color,
  marker = NULL,
  linestyle = NULL,
  replace = FALSE
)
```

Arguments

entity	Vector of strings with names of entities (regions, variable names, etc.)
legend	Vector of strings with legend names of entities.
color	Vector of strings containing hexadecimal color codes.
marker	optional Vector of strings with marker codes.
linestyle	optional Vector of strings containing linestyle codes .
replace	Logical (default FALSE) indicating whether existing data should be replaced with new data.

Author(s)

David Klein, Jan Philipp Dietrich

See Also

plotstyle

Examples

```
## Not run: plotstyle.add("AFR","Africa","#000000")
## Not run: plotstyle.add("AFR","keep","#FFFFFF")
## Not run: plotstyle.add("AFR","keep","keep",marker=20,replace=TRUE)
```

---

scenTool *scenTool*

Description

scenTool allows to explore and visualize time series of modelling results. The app is based on shiny opens in an external web brower. For more details: https://github.com/flohump/scenTool

Usage

```
scenTool(file = NULL, valfile = NULL)
```

Arguments

file	model data in CSV file in MIF format (NULL by default; in this case the user can upload files directly in the tool)
valfile	validation data in CSV file in MIF format (NULL by default; in this case the user can upload files directly in the tool)

Author(s)

Florian Humpenoeder

Examples

```
  ## Not run:
    scenTool("testdata.mif")

## End(Not run)
```

---

scenToolMAgPIE *scenToolMAgPIE*

Description

scenToolMAgPIE allows to explore and visualize time series of modelling results. The app is based on shiny opens in an external web brower. For more details: https://github.com/flohump/scenTool

Usage

```
scenToolMAgPIE(file = NULL, valfile = NULL)
```

Arguments

file	report data. Can be a CSV/MIF file or rds file with a quitte object (saved with saveRDS). file can also be a vector of rds files. NULL by default; in this case the user can upload files directly in the tool
valfile	validation data. Can be a CSV/MIF file or rds file with a quitte object (saved with saveRDS). NULL by default; in this case the user can upload files directly in the tool

Author(s)

Florian Humpenoeder

Examples

```
  ## Not run:
    scenToolMAgPIE("testdata.mif")

## End(Not run)
```

scratchBar *scratchBar*

Description

Fast visualization of a magpie object or quitte object as bar plot. If available, the years 2020 and 2050 will be selected; if not available the first and last year of the available years.

Usage

```
scratchBar(x, complete = "default", simplify = "default", ...)
```

Arguments

x	an object that can be converted to a quitte (e.g. a quitte object or a magpie objet)
complete	"default" or list with specifications for scratchComplete
simplify	"default" or list with specifications for scratchSimplify
...	furhter arguments handed on to mipBarYearData function

Value

ggplot object

Author(s)

Benjamin Leon Bodirsky

Examples

```
  ## Not run:
    x <- Intake(gdx)

## End(Not run)
```

shorten_legend *Shorten legend names to a given length*

Description

1. If ylab is specified, the function just returns everything after the ylab. 2. If identical_only = TRUE is specified, it removes identical parts in the name, independent of maxchar. 3. If maxchar is specified, the function will first try to return vector as it is, then remove elements which are identical between all elements and finally cut the end of the character vectors so that it fits the given maxchar setting. Underscores will be replaced with empty spaces for further processing

Usage

```
shorten_legend(
  x,
  maxchar = 20,
  identical_only = FALSE,
  ylab = NULL,
  sep = c(" ", "|", "_"),
  unit = NULL
)
```

Arguments

x	A character vector or a factor vector that should be shortened
maxchar	Maximum number of characters allowed
identical_only	If set to TRUE identical parts in the name will be removed, independent of the character length (maxchar will be ignored!)
ylab	If specified this part will be removed, independent of maxchar and identical_only
sep	A vector of characters which should be interpreted as separators
unit	A vector of characters with units, pasted to ylab

Author(s)

Jan Philipp Dietrich, Oliver Richters

Examples

```
a <- c("Model Scenario_BLUB", "Model Scenario_BLA", "Model Scenario_BLA_BLA_BLUB")

# do nothing
shorten_legend(a, 30)
# remove identical parts
shorten_legend(a, 15)
# or ...
shorten_legend(a, identical_only=TRUE)
# cutoff end of string
shorten_legend(a, 5)
# cut off the first part as explicitly specified
shorten_legend(a, ylab = "Model Scenario")
```

---

showAreaAndBarPlots *Show Area and Bar Plots*

Description

Creates 4 sets of plots from scenario data and shows them.

Usage

```
showAreaAndBarPlots(
  data,
  vars,
  tot = NULL,
  fill = FALSE,
  orderVars = c("mean", "user", "userRev"),
  mainReg = getOption("mip.mainReg"),
  yearsBarPlot = getOption("mip.yearsBarPlot"),
  scales = "free_y"
)
```

Arguments

data	A quitte object or an object that can be transformed into a quitte object.
vars	A character vector. The variables to be plotted.
tot	A single string. A total value to be shown in the area plots.
fill	Logical. Should the vars be normalized so that their values add to 1? (Plot shares instead of absolute values.)
orderVars	In what order should the variables be shown? "mean" Order by mean value (largest mean at bottom). The default. "user" As supplied by the user (first element of vars at top). "userRev" Reverse order of the one supplied by the user (first element of vars at bottom).
mainReg	A single string. The plots for this region are shown enlarged. Use options(mip.mainReg=<value>) to set globally.
yearsBarPlot	A numeric vector. The years shown in the bar plots. Use options(mip.yearsBarPlot=<value>) to set globally.
scales	adjusts how axes are harmonized. Default is free_y

Details

Creates 2 sets of area plots (main region + others) and 2 sets of bar plots (main region + others) of the variables specified in vars over time. For area plots, faceting is done by region and scenario; for bar plots over region. If a variables is given in tot, this is shown as a black line. If not the sum of the values of vars is drawn. If fill=TRUE, the values of vars are divided by the values of tot to show share of total. The plots arranged and shown.

Value

NULL is returned invisible.

Example Plots

Examples

```
## Not run:
options(mip.yearsBarPlot = c(2010, 2030, 2050, 2100))
options(mip.mainReg = "World")
data <- as.quitte(data)
vars <- c(
  "FE|CDR",
  "FE|Transport",
  "FE|Buildings",
  "FE|Industry")
showAreaAndBarPlots(data, vars)
showAreaAndBarPlots(data, vars, orderVars = "user")

## End(Not run)
```

`showAreaAndBarPlotsPlus` *Show Area and Bar Plots Utilizing '+'-Notation*

Description

Automatically infers disaggregation of a variable using the '+'-notation and calls `showAreaAndBarPlots`.

Usage

```
showAreaAndBarPlotsPlus(
  data,
  tot,
  plusNum = 1,
  fill = FALSE,
  mainReg = getOption("mip.mainReg"),
  yearsBarPlot = getOption("mip.yearsBarPlot"),
  scales = "free_y"
)
```

Arguments

`data`	A quitte object or an object that can be transformed into a quitte object.
`tot`	A single string. A total value to be shown in the area plots.
`plusNum`	A single number. Number of "+"symbols for disaggregation.
`fill`	Logical. Should the vars be normalized so that their values add to 1? (Plot shares instead of absolute values.)
`mainReg`	A single string. The plots for this region are shown enlarged. Use `options(mip.mainReg=<value>)` to set globally.
`yearsBarPlot`	A numeric vector. The years shown in the bar plots. Use `options(mip.yearsBarPlot=<value>)` to set globally.
`scales`	adjusts how axes are harmonized. Default is free_y

Details

The function requires data to have a character column named varplus containing variable names that use the '+'-notaion. The function searches for values in varplus that start with tot followed by "|", plusNum-times "+", and "|". These variables are then used in a call to showAreaAndBarPlots.

Value

NULL is returned invisible.

Examples

```
## Not run:
options(mip.yearsBarPlot = c(2010, 2030, 2050, 2100))
options(mip.mainReg = "World")
data <- as.quitte(data)
showAreaAndBarPlotsPlus(data, "SE|Liquids")

## End(Not run)
```

---

showLinePlots *Show Line Plots*

Description

Creates 2 sets of line plots from scenario data and shows them.

Usage

```
showLinePlots(
  data,
  vars = NULL,
  histVars = NULL,
  scales = "free_y",
  color.dim.name = NULL,
  mainReg = getOption("mip.mainReg"),
  color.dim.manual = NULL,
  histModelsExclude = NULL
)
```

Arguments

data	A quitte object or an object that can be transformed into a quitte object.
vars	A character vector. Usually just a single string. The variables to be plotted. If NULL all rows from data are plotted.
histVars	A character vector. Usually just a single string. The historical variables to be plotted. If NULL, it is set to vars.
scales	A single string. choose either "free_y" or "fixed".

color.dim.name name for the color-dimension used in the legend

mainReg A single string. The plots for this region are shown enlarged. Use options(mip.mainReg=<value>) to set globally.

color.dim.manual
optional vector with manual colors replacing default colors of color.dim, default is NULL.

histModelsExclude
A character vector with historical models to exclude. Set to NULL (default) for all available data.

Details

Two sets of line plots are shown (main region + others), depicting the values in vars over time. Faceting is done by region. The plots arranged and shown.

Value

NULL is returned invisible.

Example Plots

Examples

```
## Not run:
options(mip.mainReg = "World")
data <- as.quitte(data)
showLinePlots(data, "Policy Cost|GDP Loss")

## End(Not run)
```

---

showLinePlotsWithTarget
Show Line Plots With Target

Description

Shows line plots of a variable with additional target data.

Usage

```
showLinePlotsWithTarget(data, vars, scales = "free_y", color.dim.name = NULL)
```

Arguments

data	A quitte object or an object that can be transformed into a quitte object.
vars	A character vector. Usually just a single string. The variables to be plotted.
scales	A single string. choose either "free_y" or "fixed".
color.dim.name	name for the color-dimension used in the legend

Details

Creates a line plot showing single line plot of vars over time. Additionally target values given in variables of the form <vars>|target|<sth> are shown. The plot is shown.

Value

NULL is returned invisible.

Example Plots

Examples

```
## Not run:
data <- as.quitte(data)
showLinePlotsWithTarget(data, "Emi|GHG")

## End(Not run)
```

---

showMultiLinePlots *Show Multi-Line Plots*

Description

Show 2 sets of plots with different regions in the same plot (value over time).

Usage

```
showMultiLinePlots(
  data,
  vars,
  scales = "free_y",
  nrowNum = 1,
  mainReg = getOption("mip.mainReg")
)
```

Arguments

data	A quitte object or an object that can be transformed into a quitte object.
vars	A character vector. The variables to be plotted.
scales	A single string. choose either "free_y" or "fixed".
nrowNum	An integer value. Number of rows of the panel figures
mainReg	A single string. The plots for this region are shown enlarged. Use options(mip.mainReg=<value>) to set globally.

Details

Creates two plots (main region + others) with the values of vars over time. Different regions are shown in the same plot. Faceting is done by variable. The plots arranged and shown.

Value

NULL is returned invisible.

Example Plots

Examples

```
## Not run:
options(mip.mainReg = "World")
data <- as.quitte(data)
vars <- c(
  "FE|Transport pCap",
  "FE|Buildings pCap",
  "FE|Industry pCap")
showMultiLinePlots(data, vars)

## End(Not run)
```

---

showMultiLinePlotsByVariable

Show Multi-Line Plots by Variable

Description

Show plots with different regions in the same plot; x-axis variable chosen by user.

Usage

```
showMultiLinePlotsByVariable(
  data,
  vars,
  xVar,
  scales = "free_y",
  showHistorical = FALSE,
  showGlobal = FALSE,
  nrowNum = 1,
  mainReg = getOption("mip.mainReg"),
  histRefModel = getOption("mip.histRefModel"),
  yearsByVariable = getOption("mip.yearsBarPlot")
)
```

Arguments

data	A quitte object or an object that can be transformed into a quitte object.
vars	A character vector. The variables to be plotted.
xVar	A single string. The variable for the x-axis.
scales	A single string. choose either "free_y" or "fixed".
showHistorical	A single logical value. Should historical data be shown? It is not recommended to set this to TRUE as the resulting plot we probably be quite confusing.
showGlobal	A single logical value. Should global data be shown? Default is false to save space in pdf
nrowNum	An integer value. Number of rows of the panel figures
mainReg	A single string. The plots for this region are shown enlarged. Use options(mip.mainReg=<value>) to set globally.
histRefModel	A named character vector identifying the unique model to be chosen for historical data. Use options(mip.histRefModel=<value>) to set globally.
yearsByVariable	A numeric vector. The years to be marked in the plots. As default it uses the value globally set by options(mip.yearsBarPlot=<value>).

Details

Same as showMultiLinePlots but with the variable specified by xVar on x-axis. For every y-axis-value, we need a unique x-axis-value. For historical data, there may be several sources / models of the same variable. For the x-axis-variable a unique historical source / model is chosen via histRefModel.

Value

NULL is returned invisible.

Example Plots

Examples

```
## Not run:
options(mip.mainReg = "World")
options(mip.yearsBarPlot = c(2010, 2030, 2050, 2100))
options(mip.histRefModel = c("GDP|PPP pCap" = "James_IMF"))
data <- as.quitte(data)
vars <- c(
  "FE|Transport pCap",
  "FE|Buildings pCap",
  "FE|Industry pCap")
showMultiLinePlotsByVariable(data, vars, "GDP|PPP pCap")

## End(Not run)
```

`showRegiLinePlots` *Show Line Plots for Region Comparison*

Description

Shows line plots of a variable with different regions in the same plot and faceting by scenario.

Usage

```
showRegiLinePlots(
  data,
  vars,
  scales = "free_y",
  excludeMainRegion = TRUE,
  mainReg = getOption("mip.mainReg")
)
```

Arguments

`data`	A quitte object or an object that can be transformed into a quitte object.
`vars`	A character vector. The variables to be plotted.
`scales`	A single string. choose either `"free_y"` or `"fixed"`.
`excludeMainRegion`	A single logical value. Should the main region be excluded (or shown in the same plot)?
`mainReg`	A single string. Use options(mip.mainReg=<value>) to set globally.

Details

Creates one set of plots with the values of `vars` over time. Does not show historical data. Different regions are shown in the same plot. Faceting is done by `scenario`. The plots arranged and shown.

Value

NULL is returned invisible.

Example Plots

Examples

```
## Not run:
options(mip.mainReg = "World")
data <- as.quitte(data)
showRegiLinePlots(data, "Price|Carbon")

## End(Not run)
```

---

sideBySidePlots *sideBySidePlots*

Description

Show multiple plots side by side in one row, using the same plotly slider (frame aesthetic) if that is used.

Usage

```
sideBySidePlots(ggplots, margin = 0.05)
```

Arguments

ggplots	A list of ggplots. These plots are converted via plotly::ggplotly, however, passing plotly plots is not allowed, because they cannot be customized anymore. When using mipIterations set the argument returnGgplots to TRUE ot get ggplots instead of plotly plots.
margin	Margin between plots, passed on to plotly::subplot.

Value

A tagList containing an h3 for the titles and the ggplots in a plotly::subplot.

Author(s)

Pascal Führlich

See Also

mipIterations

theme_mip *MIP theme settings*

Description

MIP theme settings

Usage

```
theme_mip(size = 12)
```

Arguments

size	Font size

Author(s)

Jan Philipp Dietrich

Examples

```
## Not run:
  p <- mipArea(x) + theme_mip(10)

## End(Not run)
```

validationpdf *Create a validation PDF file*

Description

Create a validation PDF file

Usage

```
validationpdf(
  x,
  hist,
  file = "validation.pdf",
  style = "comparison",
  only_historical = FALSE,
  digits = 3,
  filter = NULL,
  prefix = NULL,
  hideEmptySection = FALSE,
  show_stats = TRUE,
```

```
  debug = getOption("debug"),
  pdfStyle = NULL
)
```

Arguments

x	Data to be validated. All formats allowed which can be converted to quitte (including characters containing the path to a mif or rds file)
hist	Validation data.All formats allowed which can be converted to quitte (including characters containing the path to a mif or rds file)
file	file name of the output PDF 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
style	data style for the returned data. Currently available: "trafficlight", "detailed", "comparison"
only_historical	boolean deciding whether only historical data should be used for validation or also projections from other sources
digits	integer indicating the number of digits to be shown.
filter	Additional filter to be applied on the data to only plot a subset of the provided data
prefix	Prefix which will be put in front of each part title (useful if validation is integrated into a bigger document)
hideEmptySection	removes sections in output file which would be empty for the reason that variables in input 'x' has have correspondance in the hist file
show_stats	boolean specifying whether additional statistic section should show up or not
debug	Switch to activate or deactivate debug mode.
pdfStyle	list of style-options for the pdf

Author(s)

Jan Philipp Dietrich

---

warnMissingVars *Warn If Variables Are Missing*

Description

Generates a warning if some of the variable names in vars vars are not entries of the variables-column of data.

Usage

```
warnMissingVars(data, vars)
```

Arguments

`data`	A quitte object.
`vars`	A character vector.

Value

Returns `NULL` invisibly.

Index

calculateRatio, 3
checkGlobalOptionsProvided, 4

dataframeFromGdx, 4, *7*

extractVariableGroups, 5

getLegend, 6
getPlotData, *5*, 6, *12*

harmonize, 7

identifierModelScen, 8

longestCommonPrefix, 8

mip *(*mip-package*)*, 2
mip-package, 2
mip_example_data, 14
mipArea, 9
mipBarYearData, 10
mipIterations, *7*, 11, *30*
mipLineHistorical, 12

plotPercentiles, 15
plotstyle, 16, *18*
plotstyle.add, *5*, *17*, 17

scenTool, 18
scenToolMAgPIE, 19
scratchBar, 20
shorten_legend, 20
showAreaAndBarPlots, 21, *23*, *24*
showAreaAndBarPlotsPlus, 23
showLinePlots, 24
showLinePlotsWithTarget, 25
showMultiLinePlots, 26, *28*
showMultiLinePlotsByVariable, 27
showRegiLinePlots, 29
sideBySidePlots, 30

theme_mip, 31

validationpdf, 31

warnMissingVars, 32