Package 'metis'

June 27, 2019

Title Sub-Regional Nexus Modeling Tool

Version 0.0.1

Description Package to process water-energy-land nexus data to different sub-regional levels.

Depends

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Imports raster (>= 2.7.15),
RColorBrewer (>= 1.1.2),
rgcam (>= 0.5.0),
tibble (>= 1.4.2),
ggalluvial (>= 0.9.1),
dplyr (>= 0.7.7),
tmap (>= 2.1.1),
ggplot2 (>= 3.1.0),
scales (>= 0.5.0),
utils (>= 3.5.0),
tidyr (>= 0.8.1),
rlang (>= 0.3.0),
grDevices (>= 3.5.0),
processx (>= 3.2.0),
rgdal (>= 1.2.20),
magrittr (>= 1.5),
sp (>= 1.2.7),
methods (>= 3.5.0),
tidyselect (>= 0.2.5),
rgeos (>= 0.3.26),
zoo (>= 1.8.4),
stats (>= 3.5.0),
dbplyr (>= 1.3.0),
RSQLite (>= 2.1.1),
ggrepel (>= 0.8.1),
data.table,
stringr (>= 1.3.1),
DBI

Suggests testthat (>= 2.0.1),
knitr (>= 1.20),
rmarkdown (>= 1.10)

Remotes github::JGCRI/rgcam

VignetteBuilder knitr

R topics documented:

`metis` *metis: Sub-Regional nexus Package*

Description

The Metis package provides

Metis functions

The Metis functions ...

`metis.assumptions` *metis.assumptions*

Description

This function loads holds the different assumptions used throughout the metis package.

Usage

```
metis.assumptions()
```

Details

List of Assumptions

- convEJ2TWh
- convEJ2GW
- conv1975USDperGJ22017USDperMWh
- conv1975USDperGJ22017USDperMBTU
- convertGgTgMTC
- GWPType

Value

A list of assumptions

Examples

```
library(metis)
a<-metis.assumptions()
a # will give full list of assumptions
```

`metis.bia` *metis.bia*

Description

This function downscales GCAM electricity generation and installed capacity onto a grid, based on WRI PowerWatch dataset of present capacity

Usage

```
metis.bia(biaInputsFolder = "NA", biaInputsFiles = "NA",
  reReadData = 1, regionsSelect = NULL, dataProj = "dataProj.proj",
  dataProjPath = gcamdatabasePath, scenOrigNames = NULL,
  scenNewNames = NULL, gcamdatabasePath = "NA",
  gcamdatabaseName = "NA", queryxml = "metisQueries.xml",
  queryPath = paste(getwd(), "/dataFiles/gcam", sep = ""),
  queriesSelect = "All", paramsSelect = c("elecByTech",
  "elecCapBySubsector"), gridChoice = "grid_050", diagnosticsON = T,
  subsectorNAdistribute = "even", nameAppend = "")
```

Arguments

`biaInputsFolder`	Bia Inputs Folder Path
`biaInputsFiles`	Bia Inputs Folder Path
`reReadData`	Default = 1. will read the GCAM data base and create a queryData.proj file in the same folder as the GCAM database. If FALSE will load a '.proj' file if a file with full path is provided otherwise it will search for a dataProj.proj file in the existing folder which may have been created from an old run.

regionsSelect	The regions to analyze in a vector. Example c('Colombia','Argentina')
dataProj	Optional. A default 'dataProj.proj' is produced if no .Proj file is specified.
dataProjPath	Folder that contains the dataProj or where it will be produced.
scenOrigNames	Original Scenarios names in GCAM database in a string vector. For example c('scenario1','scenario2).
scenNewNames	New Names which may be shorter and more useful for figures etc. Default will use Original Names. For example c('scenario1','scenario2)
gcamdatabasePath	Path to gcam database folder
gcamdatabaseName	Name of gcam database
queryxml	Full path to query.xml file
queryPath	Folder that contains the query.xml file.By default it is the same folder as specified by gcamdatabasePath
queriesSelect	Default = "All". Vector of queries to read from the queryxml for example
paramsSelect	Default = c("elecByTech", "elecCapBySubsector") . Vector of parameters to be read from the GCAM database
gridChoice	Default = "grid_050" . Choice of whether to use 50 km x 50 km grid cells ("grid_050") or 25 km x 25 km ("grid_025").
diagnosticsON	Default = T.
subsectorNAdistribute	Default = "even". Choose "even" for even distribution or "totalOther" to distribute based on sum of all other subsectors..
nameAppend	Default=""

Value

A tibble with GCAM electricity generation distributed on a grid for selected regions

---

`metis.boundaries` *metis.boundaries*

Description

This function takes a .csv file with gridded lat, long data and aggregates the data by spatial boundaries given different shapefiles.

Usage

```
metis.boundaries(boundaryRegShape = NULL, boundaryRegShpFolder = NULL,
  boundaryRegShpFile = NULL, boundaryRegCol = NULL,
  boundaryRegionsSelect = NULL, subRegShape = NULL,
  subRegShpFolder = NULL, subRegShpFile = NULL, subRegCol = NULL,
  subRegionsSelect = NULL, subRegType = "subRegType",
  dirOutputs = paste(getwd(), "/outputs", sep = ""), nameAppend = "",
  expandPercent = 2, overlapShape = NULL, overlapShpFolder = NULL,
  overlapShpFile = NULL, labelsSize = 1.2, fillcolorNA = NULL,
```

```
projX = "+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0",
extendedFillColor = "grey75", extendedBGColor = "lightblue1",
extendedHighLightColor = "cornsilk1", extendedLabelsColor = "grey30",
extdendedLabelSize = 0.7, extension = T, fillPalette = "Spectral",
cropSubShape2Bound = T, grids = NULL, innerMargins = c(0.1, 0.2,
0.1, 0.2), outerMargins = c(0.01, 0.01, 0.01, 0.01))
```

Arguments

boundaryRegShape	Default=NULL. Boundary region shape if already read into R.
boundaryRegShpFolder	Default= NULL. Folder containing boundary region shapefile. Suggested: paste(getwd(),"/dataFiles/g Default=""),
boundaryRegShpFile	Default=NULL. Name of shapefile. Suggested: paste("ne_10m_admin_0_countries",sep Default=""),
boundaryRegCol	Default=NULL. Column name with region names. Suggested "NAME_0",
boundaryRegionsSelect	Default=NULL. The region to choose from the given shapefile.
subRegShape	Default=NULL. Sub-region shape if already read into R.
subRegShpFolder	Default=NULL. Folder containing boundary region shapefile. Suggested paste(getwd(),"/dataFiles/gis Default=""),
subRegShpFile	Default=NULL. Name of sub-region shapefile. Suggested paste("ne_10m_admin_1_states_provinces Default=""),
subRegCol	Default= NULL. Suggested for states "name",
subRegionsSelect	Default=NULL. The region to choose from the given sub-region shapefile.
subRegType	Default="subRegType". Type of subregion. Eg. "states", "basins" etc.
dirOutputs	Default=paste(getwd(),"/outputs",sep Default=""). Location for outputs.
nameAppend	Default="". Name to append to saved files.
expandPercent	Default=2. Percentage to expand boundary region beyond chosen region.
overlapShape	Default = NULL. If boundary lines of another shapefile are desired specify the shape here.
overlapShpFolder	Default = NULL. For GCAM basins use paste(getwd(),"/dataFiles/gis/basin_gcam",sep="").
overlapShpFile	Default = NULL. For GCAM basins use ="Global235_CLM_final_5arcmin_multipart"
labelsSize	Default =1.2.
fillcolorNA	Default =NULL. Fill color for NA values.
projX	Default ="+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0".
extendedFillColor	Default = "grey75". Color used to fill extended land areas.
extendedBGColor	Default = "lightblue1". Color used to fill background/water bodies.
extendedHighLightColor	Default = "cornsilk1". Color used to highlight region of analysis.

`extendedLabelsColor`
: Default = "grey30". Color for extended country name labels.

`extdendedLabelSize`
: Default =0.7. Size of extended country name labels.

`extension`
: Default = T. Should the map be extended beyond chosen shapefile boudnaries.

`fillPalette`
: Default ="Spectral". Palette to use to fill subregions.

`cropSubShape2Bound`
: Default = T. If subregion shape file is larger than boundary file.

`grids`
: Default = NULL. Metis comes with 0.5 and 0.25 grids in c(paste(getwd(),"/dataFiles/grids/emptyGrid

`innerMargins`
: Default =c(0,0.1,0,0.1), # bottom, left, top, right

`outerMargins`
: Default =c(0.01,0.01,0.01,0.01) # bottom, left, top, right paste(getwd(),"/dataFiles/grids/emptyGrids/g
This may happen in the case of disputed boundaries.

Value

A table with data by polygon ID for each shapefile provided

`metis.chart` *metis.chart*

Description

This function produce different kinds of charts for the metis package. iIt requires a table in the Metis format. Each figure is accompanied with a csv table.

Usage

```
metis.chart(data, dataNorm = NULL, chartType = "bar",
  position = "stack", xData = "x", yData = "value",
  class = "class1", group = "scenario",
  classPalette = "classPalette1", classLabel = "classLabel1",
  color = NULL, xLabel = "xLabel", yLabel = "yLabel",
  facet_rows = NULL, facet_columns = NULL, ncolrow = 4,
  facetBGColor = "grey30", facetLabelColor = "white",
  facetLabelSize = 1.5, scales = "fixed", useNewLabels = 0,
  units = "units", xBreaksMaj = 10, xBreaksMin = 5,
  yBreaksMajn = 5, yBreaksMinn = 10, sizeBarLines = 0.5,
  sizeLines = 1.5, sectorToOrder = NULL, sectorFromOrder = NULL,
  removeCols = NULL, bubbleSize = 10, sankeyAxis1 = NULL,
  sankeyAxis2 = NULL, sankeyAxis1Label = "axis1Label",
  sankeyAxis2Label = "axis2Label", sankeyGroupColor = NULL,
  printFig = T, fileName = "chart", title = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""), figWidth = 13,
  figHeight = 9, pdfpng = "png", sankeyLabelsOn = 1,
  colOrder1 = NULL, colOrderName1 = NULL, colOrder2 = NULL,
  colOrderName2 = NULL)
```

Arguments

data	Data table for charting
dataNorm	Normalized data to plot under actual data in bubble plots. Default = NULL,
chartType	Type of chart: "bar", "line", "bubble", "sankey"
position	Position in bar charts. "identity", "stack" or "dodge"
xData	X axis data variable (dataframe or table column name). Default "x".
yData	Y axis data variable (dataframe or table column name).Default "value"
class	Class data variable (dataframe or table column name).Default "class1"
group	Group (dataframe or table column name).Default "scenario"
classPalette	Color palette to use for multiple classes. Must be a color palette eg. c("red","blue","green") or a metis.colors() palette eg. metis.colors()$pal_Basic. Default "classPalette1"
classLabel	Label to be used for legend title. Default "classLabel1"
color	A single color name for single class charts. Default NULL
xLabel	X axis title. Default "xLabel"
yLabel	Y axis title. Default "units"
facet_rows	Data variable to be used for facet rows (dataframe or table column name).Default "region"
facet_columns	Data variable to be used for facet columns (dataframe or table column name).Default "scenario"
ncolrow	Number of columns or Rows for Faceted plots.
facetBGColor	Facet background color. Default ="grey30",
facetLabelColor	Facet title text color. Default= "white",
facetLabelSize	Facet title text size. Default =1.5,
scales	Fixed or free scales for multiple sankey plots. Default "fixed"
useNewLabels	"1" or "0". Converts labels to title-case.Default 0
units	Data units. Default "units"
xBreaksMaj	X axis major breaks. Default 10
xBreaksMin	X axis minor breaks. Default 5
yBreaksMajn	Y axis major breaks. Default 5
yBreaksMinn	Y axis minor breaks. Default 10
sizeBarLines	Bar plot line size. Default 0.5
sizeLines	Line plot line size. Default 1.5
sectorToOrder	Order of "to" column variables in bubble plots. Default = NULL,
sectorFromOrder	Order of "from" column variables in bubble plots. Default = NULL,
removeCols	Option to remove certain columns from bubble plots. Default = NULL,
bubbleSize	Bubble plot bubble size. Default = 10,
sankeyAxis1	Sankey axis 1 data variable (dataframe or table column name). Default = NULL,
sankeyAxis2	Sankey axis 2 data variable (dataframe or table column name).Default = NULL,

sankeyAxis1Label
: Sankey axis 1 title data variable (dataframe or table column name).Default = "axis1Label",

sankeyAxis2Label
: Sankey axis 2 title variable (dataframe or table column name).Default = "axis2Label",

sankeyGroupColor
: Which axis variables will be used to color flow paths (One of the sankey axis). Default = NULL,

printFig
: Whether plot should be printed or not. Default = T,

fileName
: File name for plot to be saved. Default = "chart",

title
: Figure title. Default = NULL

dirOutputs
: Output directory to save figure. Default = paste(getwd(),"/outputs",sep Default = "")

figWidth
: Figure width. Default = 9,

figHeight
: Figure height. Default = 7,

pdfpng
: Whether to save plot as pdf or png. Choice between "pdf" or "png". Default = "png",

sankeyLabelsOn
: Turn on labels for sankey stratum categories. "1" or "0". Default =1

colOrder1
: Order for sankey column 1. Default = NULL,

colOrderName1
: Column name with sankey variables for column order 1. Default = NULL,

colOrder2
: Order for sankey column 1. Default = NULL,

colOrderName2
: Column name with sankey variables for column order 1. Default = NULL,

Value

Returns the formatted data used to produce chart

Examples

```
library(tibble)

# Simple example with progressively more features
tbl <- tibble::tribble (
  ~x,     ~value,
  2010,   15,
  2020,   20,
  2030,   30)

 metis.chart(data = tbl, xData = "x", yData = "value", chartType = "line")
 metis.chart(data = tbl, xData = "x", yData = "value", chartType = "bar")
 metis.chart(data = tbl, xData = "x", yData = "value", chartType = "bar", color = "blue",
            yLabel = "New y Label", xLabel = "New X label", printFig = TRUE,
            fileName = "newFileName", title = "Title")
 # More detailed data with facets
  tbl_multi <- tibble::tribble (
  ~x,     ~value, ~region,     ~scen,   ~fuel,
  2010,   25,     "region1",   "scenA",  "Oil",
  2020,   30,     "region1",   "scenA",  "Oil",
  2030,   40,     "region1",   "scenA",  "Oil",
  2010,   25,     "region2",   "scenA",  "Oil",
  2020,   10,     "region2",   "scenA",  "Oil",
```

```
2030,   60,     "region2",   "scenA",  "Oil",
2010,   75,     "region1",   "scenB",  "Oil",
2020,   30,     "region1",   "scenB",  "Oil",
2030,   20,     "region1",   "scenB",  "Oil",
2010,   25,     "region2",   "scenB",  "Oil",
2020,   10,     "region2",   "scenB",  "Oil",
2030,   90,     "region2",   "scenB",  "Oil",
2010,   55,     "region1",   "scenA",  "Gas",
2020,   40,     "region1",   "scenA",  "Gas",
2030,   30,     "region1",   "scenA",  "Gas",
2010,   35,     "region2",   "scenA",  "Gas",
2020,   30,     "region2",   "scenA",  "Gas",
2030,   32,     "region2",   "scenA",  "Gas",
2010,   16,     "region1",   "scenB",  "Gas",
2020,   28,     "region1",   "scenB",  "Gas",
2030,   39,     "region1",   "scenB",  "Gas",
2010,   12,     "region2",   "scenB",  "Gas",
2020,   26,     "region2",   "scenB",  "Gas",
2030,   37,     "region2",   "scenB",  "Gas")

my_pal <- RColorBrewer::brewer.pal(9, "Set1")

metis.chart(data = tbl_multi, xData = "x", yData = "value", class="fuel",
          chartType = "line",  classPalette=my_pal,
          facet_rows="region",facet_columns="scen")

my_pal <- metis.colors()$pal_Basic

metis.chart(data = tbl_multi, xData = "x", yData = "value", class="fuel", position="stack",
          group="fuel",chartType = "bar", classPalette=my_pal,
          facet_rows="region",facet_columns="scen")

metis.chart(data = tbl_multi, xData = "x", yData = "value", class="fuel", position="dodge",
          group="fuel",chartType = "bar", classPalette=my_pal,
          facet_rows="region",facet_columns="scen")
```

`metis.chartsProcess` *metis.chartsProcess*

Description

This function produces charts given any number of tables in the metis format. The metis.chart() function produces charts for each region nd scenario. If there are more than one scenario then the function also produces a folder for diffplots. The input tables should be .csv files with the following columns: scenario, region, sources, param, x, xLabel, vintage, class1, class2, units, value, aggregate, classLabel1,classPalette1,classLabel2,classPalette2. Running the metis.readgcam automatically produces An empty template with these columns for the relevant parameters. Each column is defined below:

Usage

```
metis.chartsProcess(dataTables = NULL, rTable = NULL, scenRef = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""), pdfpng = "png",
  xRange = "All", xCompare = c("2015", "2030", "2050", "2100"),
```

```
  paramsSelect = "All", regionsSelect = "All", xData = "x",
  yData = "value", xLabel = "xLabel", yLabel = "units",
  aggregate = "sum", class = "class", classPalette = "pal_Basic",
  regionCompareOnly = 0, scenarioCompareOnly = 0, useNewLabels = 0,
  sizeBarLines = 0, sizeLines = 1.5, nameAppend = "",
  scensSelect = "All", colOrder1 = NULL, colOrderName1 = NULL,
  colOrder2 = NULL, colOrderName2 = NULL)
```

Arguments

dataTables	Vector of strings with full path to datatables to be read in. Example c("D:/metis/outputs/Colombia/data "D:/metis/outputs/Colombia/dataTableLocal_Colombia_1975to2100.csv"). Where "dataTableLocal_Colombia_1975to2100.csv" is the new datafile created based on "dataTableTemplate_Colombia_1975to2100.csv" and contains new local data.
rTable	If a table is created directly in R as a data.frame or tibble it can entered here.
scenRef	The reference scenario to compare against. Default will pick first scenario from list f all scenarios
dirOutputs	Full path to directory for outputs. Default is paste(getwd(),"/outputs",sep="")
pdfpng	Choose the format for outputs. Either "pdf", "png" or "both. Default is "png"
xRange	Default "All". Range of x values eg. c(2001:2005)
xCompare	Choose the years to compare scenarios for xScenSelectYears plot. Default is c("2015","2030","2050","2100")
paramsSelect	Default = "All". Select the paramaters to analyze from the the tables provided. Full list of parameters: c("finalNrgbySec", "primNrgConsumByFuel", "elecByTech", "watConsumBySec", "watWithdrawBySec", "watWithdrawByCrop", "watBioPhysCons", "irrWatWithBasin","irrWatConsBasin", "gdpPerCapita", "gdp", "gdpGrowthRate", "pop", "agProdbyIrrRfd", "agProdBiomass", "agProdForest", "agProdByCrop", "landIrrRfd", "aggLandAlloc", "LUCemiss", "co2emission", "co2emissionByEndUse", "ghgEmissionByGHG", "ghgEmissByGHGGROUPS")
regionsSelect	Default = "All". Select regions to create charts for.
xData	Default "x"
yData	Default "value"
xLabel	Default "xLabel"
yLabel	Default "units"
aggregate	Default "sum"
class	Default "class"
classPalette	Default "pal_Basic" from metis.colors()$pal_Basic
regionCompareOnly	Default 0. If set to 1, will only run comparison plots and not individual
scenarioCompareOnly	Default 0. If set to 1, will only run comparison plots and not individual
useNewLabels	Default 0
sizeBarLines	Default 0.5
sizeLines	Default 1.5
nameAppend	Default =""
scensSelect	Default = "All". Select regions to create charts for.

colOrder1	Default = NULL,
colOrderName1	Default = NULL,
colOrder2	Default = NULL,
colOrderName2	Default = NULL,

Details

List of Assumptions

- scenario: The name of the new data scenario
- region: The region for the data
- sources: Sources for the data
- param: Name of the parameter
- x: The x axis variable values
- xLabel: X axis Label
- vintage: Vintages if any. If not relevant then just enter "Vintage"
- class1: Classes or types (eg. if param is water_demands then the classes may be Industry, Agriculture etc.)
- class2: A second category of classes if exists.
- units: Units for the parameter. These are used as the y axis label.
- value: The parameter value.
- aggregate: Either "sum" or "mean". This paramater is used to determine how to aggregate across regions or scenarios.
- classLabel1: If class1 exists then this will be legend Label. If it doesnt exist enter "classLabel1"
- classPalette1: An R or metis.colors() palette. Can leave the default as "pal_16".
- classLabel2: If class2 exists then this will be legend Label. If it doesnt exist enter "classLabel2"
- classPalette2: An R or metis.colors() palette. Can leave the default as "pal_16".

Value

Produces charts in output folder and also returns combined table in metis format.

`metis.colors` *metis.colors*

Description

This function loads various color palettes used previously in GCAM as well as new palettes for Metis modeling to the global environment

Usage

```
metis.colors(palx = NULL)
```

Arguments

palx	Palette name to view the palette colors. Eg. metis.colors("pal_Basic")

Details

List of Color Palettes

- pal_HDDCDD
- pal_16
- elec_tech_colors
- elec_renew_colors
- building_colors
- trn_fuel_colors
- enduse_fuel_numbered
- enduse_colors
- pal_pri_ene
- pal_pri_fuelcost
- pal_emiss_sector
- pal_landuse
- pal_hydrogen
- pal_refliq
- emiss_by_enduse_colors
- biouse_colors
- pal_Basic
- pal_Gas
- pal_Diff
- pal_Diff5
- pal_Absolute
- pal_Absolute5
- pal_Unassigned
- pal_pri_ene
- pal_nrg
- pal_hot
- pal_wet
- pal_div_wet
- pal_div_RdBl
- pal_green
- pal_div_BrGn
- pal_div_BlRd
- pal_sankey
- pal_spectral
- pal_ScarcityCat

Value

A list of color palettes.

Examples

```
library(metis)
a<-metis.colors()
pie(rep(1,length(a$pal_Basic)),label=names(a$pal_Basic),col=a$pal_Basic)
```

`metis.grid2poly` *metis.grid2poly*

Description

This function takes a .csv file with gridded lat, long data and aggregates the data by spatial boundaries given different shapefiles.

Usage

```
metis.grid2poly(grid = NULL, subRegShape = NULL,
  subRegShpFolder = NULL, subRegShpFile = NULL, subRegCol = NULL,
  subRegType = "subRegType", aggType = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""), nameAppend = "",
  labelsSize = 1.2, paramsSelect = "All", sqliteUSE = F,
  sqliteDBNamePath = paste(getwd(), "/outputs/Grids/gridMetis.sqlite",
  sep = ""))
```

Arguments

`grid`	Default=NULL. Grid file in .csv format or a R table, data frame or tibble with as a minimum columns with "lat","lon" and "value",
`subRegShape`	Default=NULL. shapefile over which grid data is to be aggregated.
`subRegShpFolder`	Default=NULL. Folder containing boundary region shapefile. Suggested paste(getwd(),"/dataFiles/gis Default=""),
`subRegShpFile`	Default=NULL. Name of sub-region shapefile. Suggested paste("ne_10m_admin_1_states_provinces Default=""),
`subRegCol`	Default= NULL. Suggested for states "name",
`subRegType`	Default="subRegType". Eg. "states", "basins" etc.
`aggType`	Default=NULL. Aggregation method to be used. Either "vol" or "depth" depen-dening on the type of data provided.
`dirOutputs`	Default=paste(getwd(),"/outputs",sep Default=""),
`nameAppend`	Default="",
`labelsSize`	Default =1.2. Label size for the region names for the gridoverlay plot.
`paramsSelect`	Default ="All"
`sqliteUSE`	Default = T,
`sqliteDBNamePath`	Default = paste(getwd(),"/outputs/Grids/gridMetis.sqlite", sep = "")

Value

A table with data by polygon ID for each shapefile provided

`metis.gridByPoly` *metis.gridByPoly*

Description

This function finds the grids located within a given shapefiles regions

Usage

```
metis.gridByPoly(grid = NULL, boundaryRegShpFolder = NULL,
  boundaryRegShpFile = NULL, colName = NULL,
  outputDir = paste(getwd(), "/outputs", sep = ""),
  fname = "gridByPoly", saveFile = F)
```

Arguments

`grid`	Default = NULL. Full path to grid file.
`boundaryRegShpFolder`	Default = NULL,
`boundaryRegShpFile`	Default = NULL,
`colName`	Default = NULL,
`outputDir`	Default = paste(getwd(),"/outputs",sep=""),
`fname`	Default = "gridByPoly"
`saveFile`	Default = F. If want csv output then change to T

Value

Prints out graphic

`metis.io` *metis.io*

Description

This function prepares gridded data for use with domestic metis modules.

Usage

```
metis.io(ioTable0 = NULL, useIntensity = 0, A0 = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""), nameAppend = "",
  figWidth = 9, figHeight = 7, sankeyLabelAbsPlots = 1,
  combSubRegionPlots = 1)
```

Arguments

ioTable0	Initial ioTable. Must have columns: supplySubSector,total,export and cap. Each supply sector should also have imports. Default = NULL,
useIntensity	Boolean to use given intensity or not. Default is set to 0.
A0	Intensity matrix. Default Null.
dirOutputs	Default =paste(getwd(),"/outputs",sep=""),
nameAppend	Modified intensity matrix. Default =NULL,
figWidth	Default = 9,
figHeight	Default = 7,
sankeyLabelAbsPlots	Default = 1
combSubRegionPlots	Default = 1

Value

A table with data by polygon ID for each shapefile provided

---

metis.map *metis.map*

Description

This function produce different kinds of maps for the metis package. Each figure is accompanied with a csv table.

Usage

```
metis.map(dataPolygon = NULL, dataGrid = NULL, dataRaster = NULL,
  shpFolder = NULL, shpFile = NULL, fillPalette = "Spectral",
  borderColor = "gray20", lwd = 1, lty = 1, bgColor = "white",
  frameShow = F, fillColumn = NULL, labels = F, labelsSize = 1.2,
  labelsColor = "black", labelsAutoPlace = T, figWidth = 9,
  figHeight = 7, legendWidth = -1, legendShow = F,
  legendOutside = F, legendTextSize = 1, legendTitleSize = 2,
  legendOutsidePosition = NULL, legendPosition = NULL,
  legendDigits = NULL, legendTitle = "Legend",
  legendStyle = "pretty", legendFixedBreaks = 5, legendBreaks = NULL,
  pdfpng = "png", underLayer = NULL, overLayer = NULL,
  printFig = T, fileName = "map", dirOutputs = paste(getwd(),
  "/outputs", sep = ""), facetFreeScale = F, facetRows = NA,
  facetCols = 3, facetBGColor = "grey30", facetLabelColor = "white",
  facetLabelSize = 1.5, alpha = 1, fillcolorNA = "gray",
  fillshowNA = NA, fillcolorNULL = "gray", facetsON = T,
  panelLabel = NULL, multiFacetRows = NULL, multiFacetCols = NULL,
  mapTitle = NULL, mapTitleSize = 1, numeric2Cat_list = NULL,
  catParam = NULL, innerMargins = c(0.01, 0.01, 0.01, 0.01),
  outerMargins = c(0.01, 0.01, 0.01, 0.01))
```

Arguments

dataPolygon	Default = NULL,
dataGrid	Default = NULL,
dataRaster	Default = NULL,
shpFolder	Default = paste(getwd(),"/dataFiles/gis/admin_gadm36_1",sep Default = ""),
shpFile	Default = paste("gadm36_1",sep Default = ""),
fillPalette	Default = "Spectral",
borderColor	Default = "gray20",
lwd	Default = 1,
lty	Default = 1,
bgColor	Default = "white",
frameShow	Default = F,
fillColumn	Default = NULL, # Or give column data with
labels	Default = F,
labelsSize	Default = 1.2,
labelsColor	Default = "black",
labelsAutoPlace	Default = F,
figWidth	Default = 9,
figHeight	Default = 7,
legendWidth	Default = -1,
legendShow	Default = F,
legendOutside	Default = T,
legendTextSize	Default = 0.8,
legendTitleSize	Default = 1,
legendOutsidePosition	Default = NULL, # "right","left","top","bottom", "center"
legendPosition	Default = NULL, # c("RIGHT','top') - RIGHT LEFT TOP BOTTOM
legendDigits	Default = NULL,
legendTitle	Default = "Legend",
legendStyle	Default = "pretty",
legendFixedBreaks	Default = "5",
legendBreaks	Default = NULL,
pdfpng	Default = "png",
underLayer	Default = NULL,
overLayer	Default = NULL,
printFig	Default = T,
fileName	Default = "map",
dirOutputs	Default = paste(getwd(),"/outputs",sep Default = ""),

facetFreeScale	Default = F,
facetRows	Default = NA,
facetCols	Default = 3,
facetBGColor	Default = "grey75",
facetLabelColor	Default = "black",
facetLabelSize	Default = 1.5,
alpha	Default = 1
fillcolorNA	Default =NULL
fillshowNA	Default =NA
fillcolorNULL	Default =NULL
facetsON	Default =F,
panelLabel	Default = NULL,
multiFacetRows	Default=NULL,
multiFacetCols	Default=NULL,
mapTitle	Default=NULL
mapTitleSize	Default=1
numeric2Cat_list	Default=NULL,
catParam	Default=NULL
innerMargins	Default =c(0,0,0,0), # bottom, left, top, right
outerMargins	Default =c(0.01,0.01,0.01,0.01) # bottom, left, top, right

Value

Returns the formatted data used to produce chart

`metis.mapProcess` *metis.mapProcess*

Description

This function produce different kinds of maps for the metis package. Each figure is accompanied with a csv table.

Usage

```
metis.mapProcess(polygonDataTables = NULL, gridDataTables = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""),
  mapsOutFolderName = "mapOutputs", xRange = "All", labels = F,
  labelsSize = 1.2, subRegShape = NULL, subRegShpFolder = NULL,
  subRegShpFile = NULL, subRegCol = NULL, subRegType = "subRegType",
  dirNameAppend = "", nameAppend = "", legendOutsideSingle = F,
  legendOutsidePosition = NULL, legendPosition = NULL,
  legendFixedBreaks = 5, legendTitleSizeO = 2, legendTextSizeO = 1,
  legendTitleSizeI = 1.5, legendTextSizeI = 1, animateOn = T,
```

```
  delay = 100, scenRef = NULL, extension = F,
  boundaryRegShape = NULL, boundaryRegShpFolder = NULL,
  boundaryRegShpFile = NULL, boundaryRegCol = NULL,
  boundaryRegionsSelect = NULL, fillcolorNA = NULL,
  extendedLabels = T, extendedFillColor = "grey75",
  extendedBGColor = "lightblue1", extendedHighLightColor = "cornsilk1",
  extendedLabelsColor = "grey30", extdendedLabelSize = 0.7,
  extendedShape = NULL, extendedShapeCol = NULL, expandPercent = 3,
  projX = "+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0",
  figWidth = 6, figHeight = 7, scaleRange = NULL,
  paramsSelect = "All", indvScenarios = "All", GCMRCPSSPPol = F,
  multiFacetCols = "scenarioRCP", multiFacetRows = "scenarioGCM",
  legendOutsideMulti = T, legendPositionMulti = NULL,
  legendTitleSizeMulti = NULL, legendTextSizeAnim = NULL,
  legendTextSizeMulti = NULL, refGCM = NULL, refRCP = NULL,
  chosenRefMeanYears = NULL, mapTitleSize = 0.5,
  facetLabelSizeMulti = 3, numeric2Cat_list = NULL, diffOn = F)
```

Arguments

polygonDataTables	Default = NULL,
gridDataTables	Default = NULL,
dirOutputs	Default = paste(getwd(),"/outputs",sep=""),
mapsOutFolderName	Default="mapOutputs",
xRange	Default ="All",
labels	Default = F,
labelsSize	Default = 1.2,
subRegShape	Default = NULL,
subRegShpFolder	Default = paste(getwd(),"/dataFiles/gis/admin_gadm36",sep=""),
subRegShpFile	Default = paste("gadm36_1",sep=""),
subRegCol	Default ="NAME_1",
subRegType	Default ="subRegType",
dirNameAppend	Default =""
nameAppend	Default =""
legendOutsideSingle	Default =F, Single plots by default have legends inside. This can be moved out if wanted.
legendOutsidePosition	Default = NULL, # "right","left","top","bottom", "center"
legendPosition	Default = NULL, # c("RIGHT','top') - RIGHT LEFT TOP BOTTOM
legendFixedBreaks	Default = "5",
legendTitleSizeO	Default = 2,
legendTextSizeO	Default =1,

legendTitleSizeI
: Default = 1,

legendTextSizeI
: Default =0.5,

animateOn
: Default = T,

delay
: Default = 100,

scenRef
: Default = NULL

extension
: Default =F,

boundaryRegShape
: Default = NULL,

boundaryRegShpFolder
: Default= NULL . Suggested paste(getwd(),"/dataFiles/gis/naturalEarth",sep Default="")

boundaryRegShpFile
: Default=NULL . Suggested paste("ne_10m_admin_0_countries",sep Default=""),

boundaryRegCol
: Default=NULL. Suggested "NAME_0",

boundaryRegionsSelect
: Default = NULL,

fillcolorNA
: Default = NULL

extendedLabels
: Default = T

extendedFillColor
: Default ="grey75",

extendedBGColor
: Default ="lightblue1",

extendedHighLightColor
: Default ="cornsilk1",

extendedLabelsColor
: Default ="grey30",

extdendedLabelSize
: Default =0.7,

extendedShape
: Default =NULL,

extendedShapeCol
: Default =NULL,

expandPercent
: Default =2

projX
: Default = projX="+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0"

figWidth
: Default =9

figHeight
: Default =7

scaleRange
: Default NULL. Dataframe with columns param, maxScale, minScale to indicate maximum and minumum values for a parameter scale.

paramsSelect
: Default ="All"

indvScenarios
: Default ="All",

GCMRCPSSPPol
: Default = F,

multiFacetCols
: Default ="scenarioRCP",

multiFacetRows
: Default ="scenarioGCM",

legendOutsideMulti
: Default = NULL,

legendPositionMulti
: Default = NULL,

legendTitleSizeMulti
: Default = NULL,

legendTextSizeAnim
: Default = NULL,

legendTextSizeMulti
: Default = NULL,

refGCM
: Default = NULL , eg. "gfdl-esm2m"

refRCP
: Default = NULL , eg. "rcp2p6"

chosenRefMeanYears
: Default=NULL

mapTitleSize
: Default=0.5

facetLabelSizeMulti
: Default =3

numeric2Cat_list
: Default=NULL,

diffOn
: Default = F. Whether to calculate diff values between scenarios.

Value

Returns the formatted data used to produce chart

---

metis.prepGrid *metis.prepGrid*

Description

This function prepares gridded data for use with other metis modules.

Usage

```
metis.prepGrid(demeterFolder = "NA", demeterScenario = "NA",
  demeterTimesteps = seq(from = 2005, to = 2100, by = 5),
  demeterUnits = "NA", tethysFolder = "NA", tethysScenario = "NA",
  tethysUnits = "NA", tethysFiles = c("wddom", "wdelec", "wdirr",
  "wdliv", "wdmfg", "wdmin", "wdnonag", "wdtotal"),
  copySingleTethysScenbyXanthos = NULL, xanthosFolder = "NA",
  xanthosFiles = "NA", xanthosScenarioAssign = "NA",
  xanthosCoordinatesPath = "NA", xanthosGridAreaHecsPath = "NA",
  scarcityXanthosRollMeanWindow = 10, spanLowess = 0.25,
  popFolder = "NA", popFiles = "NA", biaFolder = "NA",
  biaFiles = "NA", popUnits = "NA", dirOutputs = paste(getwd(),
  "/outputs", sep = ""), reReadData = 1, gridMetisData = paste(getwd(),
  "/outputs/Grids/gridMetis.RData", sep = ""), sqliteUSE = F,
  sqliteDBNamePath = paste(getwd(), "/outputs/Grids/gridMetis.sqlite",
  sep = ""))
```

Arguments

demeterFolder	Full path to demeter outputs
demeterScenario	Name of demeter scenario
demeterTimesteps	Default is seq(from=2005,to=2100,by=5)
demeterUnits	No Default
tethysFolder	Folder for tethys results
tethysScenario	Scenario name for tethys run
tethysUnits	No Default
tethysFiles	Default =c("wddom","wdelec","wdirr","wdliv","wdmfg","wdmin","wdnonag","wdtotal"),
copySingleTethysScenbyXanthos	Default=NULL,
xanthosFolder	Xanthos Folder Path
xanthosFiles	Xanthos Files to Read
xanthosScenarioAssign	Default "NA". Scenario name if testing single scenario.
xanthosCoordinatesPath	paste(getwd(),"/dataFiles/grids/xanthosCoords/coordinates.csv",sep="")
xanthosGridAreaHecsPath	=paste(getwd(),"/dataFiles/grids/xanthosRunsChris/reference/Grid_Areas_ID.csv",sep=""),
scarcityXanthosRollMeanWindow	Default = 10,
spanLowess	Default = 0.25
popFolder	Default = <-paste(getwd(),"/dataFiles/grids/griddedIDsPop/",sep="")
popFiles	Default = <-"grid_pop_map"
biaFolder	Default = <-paste(getwd(),"/dataFiles/grids/griddedIDsbia/",sep="")
biaFiles	Default = <-"grid_bia_map"
popUnits	Default = <-"person"
dirOutputs	Default =paste(getwd(),"/outputs",sep=""),
reReadData	Default =1,
gridMetisData	Default = paste(dirOutputs, "/Grids/gridMetis.RData", sep = "")
sqliteUSE	Default = T,
sqliteDBNamePath	Default = paste(getwd(),"/outputs/Grids/gridMetis.sqlite", sep = "")

Value

A table with data by polygon ID for each shapefile provided

metis.printPdfPng *metis.printPdfPng*

Description

This function prints figure to pdf or png.

Usage

```
metis.printPdfPng(figure = NULL, dir = getwd(), filename = "plot",
  figWidth = 13, figHeight = 9, pdfpng = "png")
```

Arguments

figure	Default=NULL. Figure to be printed
dir	Default = getwd(). Directory to print figure
filename	Default = "plot". File name
figWidth	Default=13.
figHeight	Default=9.
pdfpng	Default="png". Either "pdf" or "png"

Value

Prints out graphic

metis.readgcam *metis.readgcam*

Description

This function connects to a gcamdatabase and uses a query file to out results into a table ready for plotting.

Usage

```
metis.readgcam(gcamdatabasePath = NULL, gcamdatabaseName = NULL,
  queryxml = "metisQueries.xml", queryPath = paste(getwd(),
  "/dataFiles/gcam", sep = ""), scenOrigNames = NULL,
  scenNewNames = NULL, reReadData = T, dataProj = "dataProj.proj",
  dataProjPath = NULL, dirOutputs = paste(getwd(), "/outputs", sep =
  ""), regionsSelect = NULL, queriesSelect = "All",
  paramsSelect = "All")
```

Arguments

gcamdatabasePath	Path to gcam database folder
gcamdatabaseName	Name of gcam database
queryxml	Name of the query.xml file. By default it is "metisQueries.xml"
queryPath	Folder that contains the query.xml file.By default it is the same folder as specified by gcamdatabasePath
scenOrigNames	Original Scenarios names in GCAM database in a string vector. For example c('scenario1','scenario2).
scenNewNames	New Names which may be shorter and more useful for figures etc. Default will use Original Names. For example c('scenario1','scenario2)
reReadData	If TRUE will read the GCAM data base and create a queryData.proj file in the same folder as the GCAM database. If FALSE will load a '.proj' file if a file with full path is provided otherwise it will search for a dataProj.proj file in the existing folder which may have been created from an old run.
dataProj	Optional. A default 'dataProj.proj' is produced if no .Proj file is specified.
dataProjPath	Folder that contains the dataProj or where it will be produced. By default it is the same folder as specified by gcamdatabasePath
dirOutputs	Full path to directory for outputs
regionsSelect	The regions to analyze in a vector. Example c('Colombia','Argentina'). Full list: c(USA, Africa_Eastern, Africa_Northern, Africa_Southern, Africa_Western, Australia_NZ, Brazil, Canada Central America and Caribbean, Central Asia, China, EU-12, EU-15, Europe_Eastern, Europe_Non_EU, European Free Trade Association, India, Indonesia, Japan, Mexico, Middle East, Pakistan, Russia, South Africa, South America_Northern, South America_Southern, South Asia, South Korea, Southeast Asia,
queriesSelect	Default = "All". Vector of queries to read from the queryxml for example c("Total final energy by aggregate end-use sector", "Population by region"). The queries must be availble in the queryxml file. Current list of queries and generated paramaters are: • "Total final energy by aggregate end-use sector". Parameters generated: finalNrgbySec. • "primary energy consumption by region (direct equivalent)". Parameters generated: primNrgConsumByFuel • "Electricity generation by aggregate technology". Parameters generated: elecByTech • "water withdrawals by sector". Parameters generated: watWithdrawBySec • "water consumption by sector". Parameters generated: watConsumBySec • "water withdrawals by crop". Parameters generated: watWithdrawByCrop • "biophysical water demand by crop type and land region". Parameters generated: watBioPhysCons • "water withdrawals by water mapping source". Parameters generated: irrWatWithBasin • "water consumption by water mapping source". Parameters generated: irrWatConsBasin • "GDP per capita MER by region". Where MER is "Market Exchange Rate". Parameters generated: gdpPerCapita.

- "GDP MER by region". Where MER is "Market Exchange Rate". Parameters generated: gdp, gdpGrowthRate
- "Population by region". Parameters generated: pop.
- "ag production by tech". Where technologies signify irrigated or rainfed. Parameters generated: agProdbyIrrRfd
- "Ag Production by Crop Type". Parameters generated: agProdBiomass, agProdForest, agProdByCrop
- "land allocation by crop and water source". Parameters generated: landIrrRfd
- "aggregated land allocation". Parameters generated: aggLandAlloc
- "Land Use Change Emission". Parameters generated: LUCemissFut
- "CO2 Emissions by enduse". Parameters generated: co2emission, co2emissionByEndUse,
- "GHG emissions by subsector". Parameters generated: ghgEmissByGHGGROUPS, ghgEmissionByGHG

`paramsSelect` Default = "All". If desired select a subset of paramaters to analyze from the full list of parameters: c("finalNrgbySec", "primNrgConsumByFuel", "elecByTech", "watConsumBySec", "watWithdrawBySec", "watWithdrawByCrop", "watBioPhysCons", "irrWatWithBasin","irrWatConsBasin", "gdpPerCapita", "gdp", "gdpGrowthRate", "pop", "agProdbyIrrRfd", "agProdBiomass", "agProdForest", "agProdByCrop", "landIrrRfd", "aggLandAlloc", "LUCemiss", "co2emission", "co2emissionByEndUse" "ghgEmissionByGHG", "ghgEmissByGHGGROUPS")

Value

A list with the scenarios in the gcam database, queries in the queryxml file and a tibble with gcam data formatted for metis charts.

Index