Package ‘metis’

April 19, 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),
data.table,
DBI

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

Remotes github::JGCRI/rgcam

VignetteBuilder knitr

R topics documented:

`met.mrio` *metis.mrio*

Description

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

Usage

```
met.mrio(Z0 = NULL, Q0 = NULL, D0 = NULL, X0 = NULL, D = NULL,
  n_regions = 2, dirOutputs = paste(getwd(), "/outputs", sep = ""))
```

Arguments

`Z0`	Initial intermediate flow matrix. All diagnol matrices 0. Default = NULL,
`Q0`	Initial trade matrix. Columns sum to 100. Default = NULL,
`D0`	Initital External demand. Default = NULL,
`X0`	Initial total Demand internal and external. Default = NULL,
`D`	External demand or Household demand. Default = NULL,
`n_regions`	Number of regions. Default = NULL,
`dirOutputs`	Default =paste(getwd(),"/outputs",sep=""),

Value

A table with data by polygon ID for each shapefile provided

`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",
  biaScenarioAssign = "NA", zelusFolder = "NA", zelusScenario = "NA",
  zelusUnits = "NA", zelusFiles = "NA", popFolder = "NA",
  popFiles = "NA", popUnits = "NA", biaOutputsFolder = paste(getwd(),
  "/dataFiles/grids/bia/biaOutputs", sep = ""), reReadData = 1,
  gridMetisData = paste(getwd(), "/outputs/Grids/gridMetis.RData", sep =
  ""), sqliteUSE = F, sqliteDBNamePath = paste(getwd(),
  "/outputs/Grids/gridMetis.sqlite", sep = ""), regionsSelect = NULL,
  queriesSelect = "All", dataProj = gcamdataProjFile,
  scenOrigNames = c("GCAMOrig", "GCAMModified"),
  scenNewNames = c("GCAMOrig", "GCAMModified"),
  gcamdatabasePath = gcamdatabasePath,
  gcamdatabaseName = gcamdatabaseName, queryxml = "metisQueries.xml",
  paramsSelect = c("elecByTech"))
```

Arguments

`biaInputsFolder`	andym Bia Inputs Folder Path
`biaInputsFiles`	andym Bia Files to Read
`biaScenarioAssign`	andym Default "NA". Scenario name if testing a single scenario.
`zelusFolder`	andym Full path to zelus outputs
`zelusScenario`	andym Scenario name for zelus run
`zelusUnits`	andym No Default
`zelusFiles`	andym Default =c(?_?'edtrnsp','edbld','edindus'?_?)
`popFolder`	Default = <-paste(getwd(),"/dataFiles/grids/griddedIDsPop/",sep="")
`popFiles`	Default = <-"grid_pop_map"
`popUnits`	Default = <-"person"
`biaOutputsFolder`	Default =paste(getwd(),"/dataFiles/grids/bia/biaOutputs",sep=""),
`reReadData`	Default =1,
`gridMetisData`	Default = paste(dirOutputs, "/Grids/gridMetis.RData", sep = "")
`sqliteUSE`	Default = T,
`sqliteDBNamePath`	Default = paste(getwd(),"/outputs/Grids/gridMetis.sqlite", sep = "")
`regionsSelect`	The regions to analyze in a vector. Example c('Colombia','Argentina')

`queriesSelect`	Default = "All". Vector of queries to read from the queryxml for example
`dataProj`	Optional. A default 'dataProj.proj' is produced if no .Proj file is specified.
`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
`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.

Value

#andym a tibble with GCAM electricity generation distributed on a grid for a selected region

---

`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)
```

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/gi 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". Eg. "states", "basins" etc.
`dirOutputs`	Default=paste(getwd(),"/outputs",sep Default=""). Location for outputs.
`nameAppend`	Default="".
`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.
`projX`	Default ="+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0".
`extendedFillColor`	Default = "grey75".
`extendedBGColor`	Default = "lightblue1".
`extendedHighLightColor`	Default = "cornsilk1".
`extendedLabelsColor`	Default = "grey30".
`extdendedLabelSize`	Default =0.7.
`extension`	Default = T
`fillPalette`	Default ="Spectral".

`cropSubShape2Bound`	Default = T. Set to False if subregion shape is larger than boundary, but desired fro extension.
`grids`	Default = NULL. Suggested is c(paste(getwd(),"/dataFiles/grids/emptyGrids/grid_025.csv",sep=""), paste(getwd(),"/dataFiles/grids/emptyGrids/grid_050.csv",sep="")) 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, chartType = "bar", position = "stack", xData = "x",
  yData = "value", class = "class1", group = "scenario",
  classPalette = "classPalette1", classLabel = "classLabel1",
  xLabel = "xLabel", yLabel = "yLabel", facet_rows = "region",
  facet_columns = "scenario", ncolrow = 4, scales = "fixed",
  useNewLabels = 0, units = "units", xBreaksMaj = 10,
  xBreaksMin = 5, yBreaksMajn = 5, yBreaksMinn = 10,
  sizeBarLines = 0.5, sizeLines = 1.5, printFig = T,
  fileName = "chart", dirOutputs = paste(getwd(), "/outputs", sep =
  ""), figWidth = 13, figHeight = 9, pdfpng = "png")
```

Arguments

`data`	data table for charting
`chartType`	Type of chart: "bar" or "line"
`position`	Position in bar charts. "identity", "stack" or "dodge"
`xData`	Default "x"
`yData`	Default "value"
`class`	Default "class1"
`group`	Default "scenario"
`classPalette`	Default "classPalette1"
`classLabel`	Default "classLabel1"
`xLabel`	Default "xLabel"
`yLabel`	Default "units"
`facet_rows`	Default "region"
`facet_columns`	Default "scenario"

`ncolrow`	Number of columns or Rows for Faceted plots
`scales`	Default "fixed"
`useNewLabels`	Default 0
`units`	Default "units"
`xBreaksMaj`	Default 10
`xBreaksMin`	Default 5
`yBreaksMajn`	Default 5
`yBreaksMinn`	Default 10
`sizeBarLines`	Default 0.5
`sizeLines`	Default 1.5
`printFig`	Default = T,
`fileName`	Default = "map",
`dirOutputs`	Default = paste(getwd(),"/outputs",sep Default = "")
`figWidth`	Default = 9,
`figHeight`	Default = 7,
`pdfpng`	Default = "png",

Value

Returns the formatted data used to produce chart

Examples

```
# Examples below show the default chart with minimum information
# and then adding progressively more details.

library(tibble)
library(dplyr)
tbl <- 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.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 = 1, useNewLabels = 0, sizeBarLines = 0,
  sizeLines = 1.5, nameAppend = "", scensSelect = "All")
```

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
`useNewLabels`	Default 0
`sizeBarLines`	Default 0.5
`sizeLines`	Default 1.5
`nameAppend`	Default =""
`scensSelect`	Default = "All". Select regions to create charts for.

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_elec_subsec
- pal_elec_finalNrgFuel
- pal_elec_techs
- pal_elec_sec
- pal_finalNrg_sec
- pal_pri_ene
- pal_elec_tech_colors
- 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, boundaryRegionsSelect = 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",
`boundaryRegionsSelect`	Default=NULL. Larger region name which will be used as the folder name for outputs.
`subRegShape`	Default=NULL. shapefile over which grid data is to be aggregated.
`subRegShpFolder`	Default=NULL. Folder containing boundary region shapefile. Suggested paste(getwd(),"/dataFiles/gi 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" dependening 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.io` *metis.io*

Description

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

Usage

```
metis.io(Z0 = NULL, D0 = NULL, X0 = NULL, A0 = NULL,
  priorityZvsA = c(Z0 = 1, A0 = 2), priorityXvsCap = c(X0 = 1, Cap0 =
  2), Import0 = NULL, Export0 = NULL, Cap0 = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""), nameAppend = "")
```

Arguments

`Z0`	Initial Nexus Flows (i.e. Supply sectors which also have demands).Default = NULL,
`D0`	Intiial Other flows. (All other sectors which have demands but do not supply resources). Default = NULL,
`X0`	Initial Total Demands. Default = NULL,
`A0`	Initial Intensity Matrix. Default = NULL,
`priorityZvsA`	Default = c(Z0=1, A0=2),
`priorityXvsCap`	Default = c(X0=1, Cap0=2)
`Import0`	Default =NULL,
`Export0`	Default =NULL,
`Cap0`	Capacity. Default =NULL,
`dirOutputs`	Default =paste(getwd(),"/outputs",sep=""),
`nameAppend`	Modified intensity matrix. Default =NULL,

Value

A table with data by polygon ID for each shapefile provided

metis.irio *metis.irio*

Description

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

Usage

```
metis.irio(Z0 = NULL, D0 = NULL, X0 = NULL, D = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""))
```

Arguments

Z0	Default = NULL,
D0	Default = NULL,
X0	Default = NULL,
D	Default = NULL,
dirOutputs	Default =paste(getwd(),"/outputs",sep=""),

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 = F, 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 = "grey30",
  fillshowNA = NA, fillcolorNULL = "grey30", facetsON = T,
  panelLabel = NULL, multiFacetRows = NULL, multiFacetCols = NULL,
  mapTitle = NULL, mapTitleSize = 1, numeric2Cat_list = NULL,
  catParam = NULL)
```

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

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 = ""), xRange = "All",
  labels = F, labelsSize = 1.2, subRegShape = NULL,
  subRegShpFolder = NULL, subRegShpFile = NULL, subRegCol = NULL,
  subRegType = "subRegType", 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,
  extendedFillColor = "grey75", extendedBGColor = "lightblue1",
  extendedHighLightColor = "cornsilk1", extendedLabelsColor = "grey30",
  extdendedLabelSize = 0.7, extendedShape = NULL,
  extendedShapeCol = NULL, expandPercent = 2,
  projX = "+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0",
  figWidth = 9, figHeight = 7, scaleRange = NULL,
  paramsSelect = "All", indvScenarios = NULL, 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)
```

Arguments

`polygonDataTables`	Default = NULL,
`gridDataTables`	Default = NULL,
`dirOutputs`	Default = paste(getwd(),"/outputs",sep=""),
`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",
`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
`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 =T,

`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,

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",
  biaFolder = "NA", biaFiles = "NA", biaScenarioAssign = "NA",
  zelusFolder = "NA", zelusScenario = "NA", zelusUnits = "NA",
  zelusFiles = "NA", scarcityXanthosRollMeanWindow = 10,
  spanLowess = 0.25, popFolder = "NA", popFiles = "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=""),
`biaFolder`	andym Bia Folder Path
`biaFiles`	andym Bia Files to Read
`biaScenarioAssign`	andym Default "NA". Scenario name if testing a single scenario.
`zelusFolder`	andym Full path to zelus outputs
`zelusScenario`	andym Scenario name for zelus run
`zelusUnits`	andym No Default
`zelusFiles`	andym Default =c(?_?'edtrnsp','edbld','edindus'?_?)
`scarcityXanthosRollMeanWindow`	Default = 10,
`spanLowess`	Default = 0.25
`popFolder`	Default = <-paste(getwd(),"/dataFiles/grids/griddedIDsPop/",sep="")
`popFiles`	Default = <-"grid_pop_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.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, gcamdatabaseName,
  queryxml = "metisQueries.xml", queryPath = gcamdatabasePath,
  scenOrigNames, scenNewNames = NULL, reReadData = T,
  dataProj = "dataProj.proj", dataProjPath = gcamdatabasePath,
  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')
`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.

`metis.templates` *metis.templates*

Description

This script holds various templates used for different scripts.

Usage

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

metis.chartsThemeLight()
```

Arguments

`figure`	Figure to be printed in function metis.printPdfPng
`dir`	Directory to print figure to in function metis.printPdfPng
`filename`	Filename for figure printed in function metis.printPdfPng
`figWidth`	Figure Width in inches for figures to be printed in function metis.printPdfPng
`figHeight`	Figure height in inches for figures to be printed in function metis.printPdfPng
`pdfpng`	Either "pdf", "png" or "both" to define the format of output

Details

List of Templates in this script:

- metis.printPdfPng: Function used to print charts to a pdf or png or both.
- metis.chartsThemeLight: A light ggplot theme for charts
- metis.tmapAnimate: A function to animate tmaps across a variable.
- metis.tmapLayout: A fucntion to define tmap layouts

Value

A list of different templates

Index