

# Package ‘metis’

December 13, 2018

**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

**Suggests** testthat, knitr, rmarkdown

**RoxygenNote** 6.1.0

**Imports**

raster, RColorBrewer, rgcam, tibble, dplyr, tmap, ggplot2, scales, utils, tidyr, rlang, grDevices, processx, rgdal, magrittr, sp, methods, tidyselect, rgeos, zoo, stats

**Remotes** github::JGCRI/rgcam

**VignetteBuilder** knitr

## R topics documented:

metis . . . . .	2
metis.assumptions . . . . .	2
metis.chart . . . . .	3
metis.chartsProcess . . . . .	4
metis.colors . . . . .	6
metis.grid2poly . . . . .	8
metis.map . . . . .	9
metis.mapProcess . . . . .	11
metis.prepGrid . . . . .	12
metis.readgcam . . . . .	13
metis.templates . . . . .	15
<b>Index</b>	<b>16</b>

---

metis	<i>metis: Sub-Regional nexus Package</i>
-------	--

---

## Description

The Metis package provides

## Metis functions

The Metis functions ...

---

metis.assumptions	<i>metis.assumptions</i>
-------------------	--------------------------

---

## 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.chart

*metis.chart*

## Description

This function produce different kinds of charts for the metis package. It 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, yBreaksMaj = 5, yBreaksMin = 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
yBreaksMaj	Default 5
yBreaksMin	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	<i>metis.chartsProcess</i>
---------------------	----------------------------

---

### Description

This function produces charts given any number of tables in the metis format. The metis.chart() function produces charts for each region and 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 = "")
```

## Arguments

<code>dataTables</code>	Vector of strings with full path to datatables to be read in. Example <code>c("D:/metis/outputs/Colombia/dataTable.Colombia_1975to2100.csv", "D:/metis/outputs/Colombia/dataTableLocal.Colombia_1975to2100.csv")</code> . Where "dataTableLocal.Colombia_1975to2100.csv" is the new datafile created based on "dataTableTemplate.Colombia_1975to2100.csv" and contains new local data.
<code>rTable</code>	If a table is created directly in R as a data.frame or tibble it can entered here.
<code>scenRef</code>	The reference scenario to compare against. Default will pick first scenario from list of all scenarios
<code>dirOutputs</code>	Full path to directory for outputs. Default is <code>paste(getwd(), "/outputs", sep="")</code>
<code>pdfpng</code>	Choose the format for outputs. Either "pdf", "png" or "both". Default is "png"
<code>xRange</code>	Default "All". Range of x values eg. <code>c(2001:2005)</code>
<code>xCompare</code>	Choose the years to compare scenarios for xScenSelectYears plot. Default is <code>c("2015", "2030", "2050", "2100")</code>
<code>paramsSelect</code>	Default = "All". Select the parameters to analyze from the tables provided. Full list of parameters: <code>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")</code>
<code>regionsSelect</code>	Default = "All". Select regions to create charts for.
<code>xData</code>	Default "x"
<code>yData</code>	Default "value"
<code>xLabel</code>	Default "xLabel"
<code>yLabel</code>	Default "units"
<code>aggregate</code>	Default "sum"
<code>class</code>	Default "class"
<code>classPalette</code>	Default "pal_Basic" from <code>metis.colors()</code> \$pal_Basic
<code>regionCompareOnly</code>	Default 0. If set to 1, will only run comparison plots and not individual
<code>useNewLabels</code>	Default 0
<code>sizeBarLines</code>	Default 0.5
<code>sizeLines</code>	Default 1.5
<code>nameAppend</code>	Default ""

## 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	<i>metis.colors</i>
--------------	---------------------

---

### 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\_reffiq
- 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

**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, boundaryRegShape = NULL,
  boundaryRegShpFolder = paste(getwd(), "/dataFiles/gis/admin_gadm36",
    sep = ""), boundaryRegShpFile = paste("gadm36_0", sep = ""),
  boundaryRegCol = "NAME_0", boundaryRegionsSelect = NULL,
  subRegShape = NULL, subRegShpFolder = paste(getwd(),
    "/dataFiles/gis/admin_gadm36", sep = ""),
  subRegShpFile = paste("gadm36_1", sep = ""), subRegCol = "NAME_1",
  subRegionsSelect = NULL, subRegType = "subRegType", aggType = NULL,
  dirOutputs = paste(getwd(), "/outputs", sep = ""), nameAppend = "",
  expandbboxPercent = 2, extension = F, overlapShpFolder = NULL,
  overlapShpFile = NULL)
```

## Arguments

```
grid           Default=NULL,
boundaryRegShape
               Default=NULL,
boundaryRegShpFolder
               Default=paste(getwd(),"/dataFiles/gis/admin_gadm36",sep Default=""),
boundaryRegShpFile
               Default=paste("gadm36_0",sep Default=""),
boundaryRegCol Default="NAME_0",
boundaryRegionsSelect
               Default=NULL,
subRegShape    Default=NULL,
subRegShpFolder
               Default=paste(getwd(),"/dataFiles/gis/admin_gadm36",sep Default=""),
subRegShpFile  Default=paste("gadm36_1",sep Default=""),
subRegCol      Default="NAME_1",
subRegionsSelect
               Default=NULL,
subRegType     Default="subRegType",
aggType        Default=NULL,
dirOutputs     Default=paste(getwd(),"/outputs",sep Default=""),
nameAppend     Default="",
expandbboxPercent
               Default=2,
```



extension           Default=T,  
 overlapShpFolder           Default = NULL. Can save to paste(getwd(),"/dataFiles/gis/basin\_gcam",sep=""),  
 overlapShpFile   Default = NULL. Can save to ="Global235.CLM\_final\_5arcmin\_multipart"

### Value

A table with data by polygon ID for each shapefile provided

---

metis.map	<i>metis.map</i>
-----------	------------------

---

### 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 = paste(getwd(), "/dataFiles/gis/admin_gadm36", sep = ""),
  shpFile = paste("gadm36_1", sep = ""), shpName = "NAME_0",
  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 = T,
  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, facetLabelColor = "grey75", facetLabelSize = 1.5,
  alpha = 1, rasterCoverNegShape = F)
```

### 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 = """),  
 shpName           Default = "NAME\_0",  
 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 = ""),
facetFreeScale     Default = F,
facetRows          Default = NA,
facetCols          Default = 3,
facetLabelColor    Default = "grey75",
facetLabelSize     Default = 1.5,
alpha              Default = 1
rasterCoverNegShape Default = T

```

## Value

Returns the formatted data used to produce chart

---

metis.mapProcess	<i>metis.mapProcess</i>
------------------	-------------------------

---

## 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, regionsSelect = NULL,
  subRegShape = NULL, subRegShpFolder = paste(getwd(),
    "/dataFiles/gis/admin_gadm36", sep = ""),
  subRegShpFile = paste("gadm36_1", sep = ""), subRegCol = "NAME_1",
  subRegType = "subRegType", nameAppend = "",
  rasterCoverNegShape = T, legendOutsidePosition = NULL,
  legendPosition = NULL, legendFixedBreaks = 5, animateOn = T,
  delay = 100, legendTitleSize = 1, scenRef = NULL)
```

## Arguments

polygonDataTables	Default = NULL,
gridDataTables	Default = NULL,
dirOutputs	Default = paste(getwd(), "/outputs", sep = ""),
xRange	Default = "All",
labels	Default = F,
labelsSize	Default = 1.2,
regionsSelect	Default = NULL,
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 = ""
rasterCoverNegShape	Default = T
legendOutsidePosition	Default = NULL, # "right", "left", "top", "bottom", "center"
legendPosition	Default = NULL, # c("RIGHT", "top") - RIGHT LEFT TOP BOTTOM
legendFixedBreaks	Default = "5",
animateOn	Default = T,

delay            Default = 100,  
 legendTitleSize            Default = 1,  
 scenRef            Default = NULL

### Value

Returns the formatted data used to produce chart

---

metis.prepGrid	<i>metis.prepGrid</i>
----------------	-----------------------

---

### Description

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

### Usage

```
metis.prepGrid(demeterFolder = NULL, demeterScenario = NULL,
  demeterTimesteps = seq(from = 2005, to = 2100, by = 5),
  demeterUnits = NULL, tethysFolder = NULL, tethysScenario = NULL,
  tethysUnits = NULL, tethysFiles = c("wddom", "wdelec", "wdirr",
    "wdliv", "wdmfg", "wdmin", "wdnonag", "wdtotal"), xanthosFolder = NULL,
  xanthosScenario = NULL, xanthosUnits = NULL, xanthosFiles = NULL,
  xanthosCoordinatesPath = paste(getwd(),
    "/dataFiles/grids/xanthosCoords/coordinates.csv", sep = ""),
  scarcityXanthosRollMeanWindow = 10, dirOutputs = paste(getwd(),
    "/outputs", sep = ""), reReadData = 1,
  gridMetisData = paste(dirOutputs, "/Grids/gridMetis.RData", 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"),  
 xanthosFolder    Xanthos Folder Path  
 xanthosScenario            Xanthos Scenario Name  
 xanthosUnits    Xanthos Untis  
 xanthosFiles    Xanthos Files to Read  
 xanthosCoordinatesPath            paste(getwd(), "/dataFiles/grids/xanthosCoords/coordinates.csv", sep = "")

```

scarcityXanthosRollMeanWindow
                        Default = 10,
dirOutputs             Default = paste(getwd(), "/outputs", sep = ""),
reReadData             Default = 1,
gridMetisData          Default = paste(dirOutputs, "/Grids/gridMetis.RData", sep = "")

```

### Value

A table with data by polygon ID for each shapefile provided

---

metis.readgcam	<i>metis.readgcam</i>
----------------	-----------------------

---

### 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", scenOrigNames, scenNewNames = NULL,
               reReadData = T, dataProj = "dataProj.proj",
               dirOutputs = paste(getwd(), "/outputs", sep = ""),
               regionsSelect = NULL, queriesSelect = "All", paramsSelect = "All")

```

### Arguments

gcamdatabasePath	Path to gcam database folder
gcamdatabaseName	Name of gcam database
queryxml	Full path to query.xml file
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.
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 available 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: ghgEmissionByGHGGROUPS, ghgEmissionByGHG

**paramsSelect** Default = "All". If desired select a subset of parameters 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", "ghgEmissionByGHGGROUPS")

## 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	<i>metis.templates</i>
-----------------	------------------------

---

## 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

<i>figure</i>	Figure to be printed in function metis.printPdfPng
<i>dir</i>	Directory to print figure to in function metis.printPdfPng
<i>filename</i>	Filename for figure printed in function metis.printPdfPng
<i>figWidth</i>	Figure Width in inches for figures to be printed in function metis.printPdfPng
<i>figHeight</i>	Figure height in inches for figures to be printed in function metis.printPdfPng
<i>pdfpng</i>	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 function to define tmap layouts

## Value

A list of different templates

# Index

- \*Topic **assumptions**
  - metis.assumptions, 2
- \*Topic **charts**,
  - metis.chart, 3
  - metis.chartsProcess, 4
  - metis.map, 9
  - metis.mapProcess, 11
  - metis.templates, 15
- \*Topic **colors**,
  - metis.colors, 6
- \*Topic **database**,
  - metis.grid2poly, 8
  - metis.prepGrid, 12
  - metis.readgcam, 13
- \*Topic **diffplots**
  - metis.chart, 3
  - metis.chartsProcess, 4
  - metis.map, 9
  - metis.mapProcess, 11
- \*Topic **gcam**,
  - metis.grid2poly, 8
  - metis.prepGrid, 12
  - metis.readgcam, 13
- \*Topic **gcam**
  - metis.grid2poly, 8
  - metis.prepGrid, 12
  - metis.readgcam, 13
- \*Topic **maps**,
  - metis.templates, 15
- \*Topic **palette**
  - metis.colors, 6
- \*Topic **print**
  - metis.templates, 15
- \*Topic **query**
  - metis.grid2poly, 8
  - metis.prepGrid, 12
  - metis.readgcam, 13
- \*Topic **templates**,
  - metis.templates, 15

metis, 2

metis-package (metis), 2

metis.assumptions, 2

metis.chart, 3

metis.chartsProcess, 4

metis.chartsThemeLight (metis.templates), 15

metis.colors, 6

metis.grid2poly, 8

metis.map, 9

metis.mapProcess, 11

metis.prepGrid, 12

metis.printPdfPng (metis.templates), 15

metis.readgcam, 13

metis.templates, 15