Package ‘foreSIGHT’

December 4, 2019

Version 0.9.81

Depends R (>= 3.3.0),GA (>= 3.0.2), zoo

Imports doParallel, ggplot2 (>= 3.0.0), directlabels, cowplot, stats, graphics, grDevices, utils, moments

Suggests knitr (>= 1.8), rmarkdown (>= 1.18)

Title Systems Insights from Generation of Hydroclimatic Timeseries

Author Bree Bennett [aut, cre] (<https://orcid.org/0000-0002-2131-088X>),
Sam Culley [aut] (<https://orcid.org/0000-0003-4798-8522>),
Seth Westra [aut] (<https://orcid.org/0000-0003-4023-6061>),
Danlu Guo [ctb] (<https://orcid.org/0000-0003-1083-1214>),
Holger Maier [ths] (<https://orcid.org/0000-0002-0277-6887>)

Maintainer Bree Bennett <bree.bennett@adelaide.edu.au>

BugReports https://github.com/bsbennett/foreSIGHT/issues

Description A tool to create hydroclimate scenarios, stress test systems and visualize system performance in scenario-neutral climate change impact assessments. Scenario-neutral approaches 'stress-test' the performance of a modelled system by applying a wide range of plausible hydroclimate conditions (see Brown & Wilby (2012) <doi:10.1029/2012EO410001> and Prudhomme et al. (2010) <doi:10.1016/j.jhydrol.2010.06.043>). These approaches allow the identification of hydroclimatic variables that affect the vulnerability of a system to hydroclimate variation and change. This tool enables the generation of perturbed time series using a range of approaches including simple scaling of observed time series (e.g. Culley et al. (2016) <doi:10.1002/2015WR018253>) and stochastic simulation of perturbed time series via an inverse approach (see Guo et al. (2018) <doi:10.1016/j.jhydrol.2016.03.025>). It incorporates a number of stochastic weather models to generate hydroclimate variables on a daily basis (e.g. precipitation, temperature, potential evapotranspiration) and allows a variety of different hydroclimate variable properties, herein called attributes, to be perturbed. Options are included for the easy integration of existing system models both internally in R and externally for seamless 'stress-testing'. A suite of visualization options for the results of a scenario-neutral analysis (e.g. plotting performance spaces and overlaying climate projection information) are also included. As further developments in scenario-neutral approaches occur the tool will be updated to incorporate these advances.

License GPL-3

NeedsCompilation no

VignetteBuilder knitr

Repository CRAN

Date/Publication 2019-12-04 06:30:04 UTC

R topics documented:

`attributeCalculator` *attributeCalculator*

Description

Calculates the attributes of a supplied set of time series data.

Usage

```
attributeCalculator(obs = NULL,
                    attSel = NULL,
                    slice = NULL,
                    window = 10
                    )
```

Arguments

`obs`	A dataframe of observed climate data in the form *Year Month Day P Temp*.
`attSel`	A character vector of climate attributes selected for calculation. A list of all supported attributes can be found in details below.
`slice`	Optional argument. A scalar value indicating the last year (e.g. 2050) of the supplied time series that is used as in input for attribute calcualtion.
`window`	the period (number of years) over which the attributes are calcualted. Note: the window is the number of years taken before the slice year. Default set at 10.

Details

The list of attributes supported by attSel are:

- "P_ann_tot_m"
- "P_ann_R10_m"
- "P_ann_maxDSD_m"
- "P_ann_maxWSD_m"
- "P_ann_P99_m"
- "P_ann_dyWet99p_m"
- "P_ann_ratioWS_m"
- "Temp_ann_avg_m"
- "Temp_ann_P5_m"
- "Temp_ann_P95_m"
- "Temp_ann_F0_m"
- "P_ann_dyWet_m"
- "P_ann_DSD_m"
- "P_seas_tot_cv"
- "P_mon_tot_cv"
- "P_ann_avgWSD_m"
- "P_ann_avgDSD_m"
- "P_JJA_avgWSD_m"
- "P_MAM_avgWSD_m"
- "P_DJF_avgWSD_m"
- "P_SON_avgWSD_m"
- "P_JJA_avgDSD_m"
- "P_MAM_avgDSD_m"
- "P_DJF_avgDSD_m"
- "P_SON_avgDSD_m"
- "Temp_ann_GSL_m"
- "Temp_ann_CSL_m"

- "P_JJA_dyWet_m"
- "P_MAM_dyWet_m"
- "P_DJF_dyWet_m"
- "P_SON_dyWet_m"
- "P_JJA_tot_m"
- "P_MAM_tot_m"
- "P_DJF_tot_m"
- "P_SON_tot_m"
- "P_ann_nWet_m"
- "P_ann_dyAll_m"
- "P_JJA_dyAll_m"
- "P_MAM_dyAll_m"
- "P_DJF_dyAll_m"
- "P_SON_dyAll_m"
- "PET_ann_avg_m"
- "PET_ann_tot_m"
- "PET_ann_rng_m"
- "Temp_ann_rng_m"
- "PET_ann_90pX_m"
- "P_ann_90X_m"
- "P_ann_seasRatio_m"
- "PET_ann_P5_m"
- "PET_ann_P95_m"
- "P_Jan_tot_m"
- "P_Feb_tot_m"
- "P_Mar_tot_m"
- "P_Apr_tot_m"
- "P_May_tot_m"
- "P_Jun_tot_m"
- "P_Jul_tot_m"
- "P_Aug_tot_m"
- "P_Sep_tot_m"
- "P_Oct_tot_m"
- "P_Nov_tot_m"
- "P_Dec_tot_m"
- "Temp_JJA_avg_m"
- "Temp_MAM_avg_m"

- "Temp_DJF_avg_m"
- "Temp_SON_avg_m"
- "Temp_Jan_avg_m"
- "Temp_Feb_avg_m"
- "Temp_Mar_avg_m"
- "Temp_Apr_avg_m"
- "Temp_May_avg_m"
- "Temp_Jun_avg_m"
- "Temp_Jul_avg_m"
- "Temp_Aug_avg_m"
- "Temp_Sep_avg_m"
- "Temp_Oct_avg_m"
- "Temp_Nov_avg_m"
- "Temp_Dec_avg_m"
- "PET_JJA_avg_m"
- "PET_MAM_avg_m"
- "PET_DJF_avg_m"
- "PET_SON_avg_m"
- "PET_JJA_tot_m"
- "PET_MAM_tot_m"
- "PET_DJF_tot_m"
- "PET_SON_tot_m"
- "PET_Jan_tot_m"
- "PET_Feb_tot_m"
- "PET_Mar_tot_m"
- "PET_Apr_tot_m"
- "PET_May_tot_m"
- "PET_Jun_tot_m"
- "PET_Jul_tot_m"
- "PET_Aug_tot_m"
- "PET_Sep_tot_m"
- "PET_Oct_tot_m"
- "PET_Nov_tot_m"
- "PET_Dec_tot_m"
- "PET_Jan_avg_m"
- "PET_Feb_avg_m"
- "PET_Mar_avg_m"

- "PET_Apr_avg_m"
- "PET_May_avg_m"
- "PET_Jun_avg_m"
- "PET_Jul_avg_m"
- "PET_Aug_avg_m"
- "PET_Sep_avg_m"
- "PET_Oct_avg_m"
- "PET_Nov_avg_m"
- "PET_Dec_avg_m"
- "PET_ann_seasRatio_m"
- "Radn_ann_avg_m"
- "Radn_ann_tot_m"
- "Radn_ann_rng_m"
- "Radn_ann_P5_m"
- "Radn_ann_P95_m"
- "Radn_JJA_avg_m"
- "Radn_MAM_avg_m"
- "Radn_DJF_avg_m"
- "Radn_SON_avg_m"
- "Radn_JJA_tot_m"
- "Radn_MAM_tot_m"
- "Radn_DJF_tot_m"
- "Radn_SON_tot_m"
- "Radn_Jan_tot_m"
- "Radn_Feb_tot_m"
- "Radn_Mar_tot_m"
- "Radn_Apr_tot_m"
- "Radn_May_tot_m"
- "Radn_Jun_tot_m"
- "Radn_Jul_tot_m"
- "Radn_Aug_tot_m"
- "Radn_Sep_tot_m"
- "Radn_Oct_tot_m"
- "Radn_Nov_tot_m"
- "Radn_Dec_tot_m"
- "Radn_Jan_avg_m"
- "Radn_Feb_avg_m"

- "Radn_Mar_avg_m"
- "Radn_Apr_avg_m"
- "Radn_May_avg_m"
- "Radn_Jun_avg_m"
- "Radn_Jul_avg_m"
- "Radn_Aug_avg_m"
- "Radn_Sep_avg_m"
- "Radn_Oct_avg_m"
- "Radn_Nov_avg_m"
- "Radn_Dec_avg_m"
- "Radn_ann_seasRatio_m"

Examples

```
###Example 1 - calculate attributes for full length of data
data(tankDat)
attSel=c("P_ann_tot_m","P_ann_dyWet_m","P_ann_dyAll_m")
out=attributeCalculator(obs = tank_obs,
                        attSel = attSel,
                        slice=NULL,
                        window=NULL)
```

---

climdata *Example climate projection data*

Description

A dataframe of climate projection data for superposition on performance spaces via plotLayers

Format

climdata is a dataframe with 12 rows and 3 columns

climdata A dataframe of climate attributes and performance in the form *P_ann_tot_m Temp_ann_avg_m performance*.

devScenarioClean *devScenarioClean*

Description

Takes the IO="dev" output of scenariogenerator and repacks it into the IO="verbose" style.

Usage

```
devScenarioClean(obs=NULL,
                 optimArgs=NULL,
                 attPerturb=NULL,
                 attHold=NULL,
                 attPenalty=NULL,
                 modelTag=NULL,
                 modelInfoMod=list(),
                 exSpArgs=NULL,
                 simDirectory=NULL,
                 nSeed=NULL,
                 nRep=NULL,
                 repack=FALSE
                 )
```

Arguments

`obs` A dataframe of observed climate data in the form *Year Month Day P Temp*.

`optimArgs` A list controlling the search algorithm with the following components:

- `pcrossover` a value of probability of crossover. Defaults to 0.8.
- `pmutation` a value of probability of mutation. Defaults to 0.1
- `maxiter` a value of the maximum number of generations. Defaults to 50
- `maxFitness` a value of the stopping criteria. Defaults to 0.001
- `popSize` a value of the population size. Defaults to 100
- `run` a value of an alternative stopping criteria - consecutive runs without improvement in fitness. Defaults to 0.001
- `seed` a value of the random seed. Defaults to 1234.
- `parallel` specifies if parallel computing should be used. Defaults to False. Can be set to the number of desired cores, or `TRUE`, where it will detect the number of available cores and run.
- `keepBest` specifies if the optimisation should keep the best solution in each generation. Defaults to TRUE.
- `lambda.mult` A multiplier used during the optimisation for primary attributes. Defaults to zero.

`attPerturb` A character vector of climate attributes to hold at a target. A list of all supported attributes can be found under shown under details below.

`attHold` A character vector of climate attributes to hold at a target. A list of all supported attributes can be found under shown under details below.

`attPenalty` A character vector of climate attributes to place specific focus on during targeting via the use of a penalty function during the optimisation process.

`modelTag` A character vector of which stochastic models to use to create each climate variable.Supported tags are shown under details below.

`modelInfoMod` A list containing information for modifying stochastic model bounds and defining fixed parameters.

`exSpArgs` a list to control the exposure space creation with the following components:

- `type` a string that specifies the type of sampling. Defaults to regular spacing.
- `samp` a vector indicating the number of targets for each attribute in attSel.
- `bounds` a list containing elements for attributes listed in attSel, where each attribute has bounds specified. This should be a single value for a stationary target, and a vector of min and max change for primary attributes. Works with samp to create number of step sizes. Defaults with samp to only reproduce historical weather.

`simDirectory` A string used to label the output directory.

`nSeed` A scalar used to specify the number of seeds (in this case replicates) for the stochastic generation of time series.

`nRep` A scalar that indicates the total number of files (i.e. no. targets x n seeds)

`repack` If TRUE writes .csv files of all collates scenarios.

Details

Repackages output from IOmode="dev" runs of sceanariogenerator function.

`exSpArgsVisual` *exSpArgsVisual*

Description

Visualizes the geometry of a 2D exposure space. This visualizer only works for 2d spaces (samples of 2 attributes).

Usage

```
exSpArgsVisual(exSpArgs=NULL
                     )
```

Arguments

exSpArgs — a list to control the exposure space creation with the following components:

- `type` a string that specifies the type of sampling. Defaults to regular spacing.
- `samp` a vector indicating the number of targets for each attribute in attSel.
- `bounds` a list containing elements for attributes listed in attSel, where each attribute has bounds specified. This should be a single value for a stationary target, and a vector of min and max change for primary attributes. Works with samp to create number of step sizes. Defaults with samp to only reproduce historical weather.

Examples

```
library(foreSIGHT)              ###Load package

###Example 1 - visualize a 2D exposure space.
exSpArgs=list(type="regGrid",
              samp=c(5,7),
              bounds=list(P_ann_tot_m=c(0.7,1.3),
                          Temp_ann_avg_m=c(-3,3)))
#windows()
exSpArgsVisual(exSpArgs=exSpArgs)
```

---

modCalibrator — *modCalibrator*

Description

Calibrates weather generator models specified using modelTag.

Usage

```
modCalibrator(obs = NULL,
              modelTag = NULL)
```

Arguments

`obs`	A dataframe of observed climate data in the form *Year Month Day P Temp*.
`modelTag`	A character vector of which stochastic models to use to create each climate variable. Supported tags are shown in under details below.

Details

modelTag provides the main function with requested models. modelTag is vector of any of the following supported models:

- `"Simple-ann"`a simple annual scaling

- "P-ann-wgen"a four parameter annual rainfall model
- "P-seas-wgen"a 16 parameter seasonal rainfall model
- "P-har6-wgen"a harmonic rainfall model with 6 periods
- "P-har12-wgen"a harmonic rainfall model
- "P-har12-wgen-FS"a harmonic rainfall model where seasonality is fixed (phase angles must be specified via modelInfoMod=list("P-har12-wgen-FS"=fixedPars=c(x,x,x,x))
- "P-har26-wgen"a harmonic rainfall model
- "P-2har26-wgen"a double harmonic rainfall model
- "Temp-har26-wgen"a harmonic temperature model not conditional on rainfall
- "Temp-har26-wgen-wd"a harmonic temperature model dependent on wet or dry day
- "Temp-har26-wgen-wdsd"a harmonic temperature model where standard deviation parameters are dependent on wet or dry day
- "PET-har12-wgen"a harmonic potential evapotranspiration model
- "PET-har26-wgen"a harmonic potential evapotranspiration model
- "PET-har26-wgen-wd"a harmonic potential evapotranspiration model dependent on wet or dry day
- "Radn-har26-wgen"a harmonic solar radiation model (MJ/m2)

Examples

```
data(tankDat)                                  #Load tank data (tank_obs)
modelTag=c("P-ann-wgen","Temp-har26-wgen")  #Select a rainfall and a temperature generator
out<- modCalibrator(obs = tank_obs,          #Calibrate models
                    modelTag = modelTag)
```

---

modSimulator *modSimulator*

Description

Simulates using weather generator models specified using modelTag.

Usage

```
modSimulator(datStart=NULL,
             datFinish=NULL,
             modelTag=NULL,
             parS=NULL,
             seed=NULL,
             file=NULL,
             IOmode="suppress"
             )
```

Arguments

`datStart`	A date string in an accepted date format e.g. *"01-10-1990"*.
`datFinish`	A date string in an accepted date format e.g. *"01-10-1990"*. Must occur after datStart.
`modelTag`	A character vector of which stochastic models to use to create each climate variable. Supported tags are shown in details below.
`parS`	A list (names must match supplied modelTags) containing numeric vectors of model parameters.
`seed`	Numeric. Seed value supplied to weather generator.
`file`	Character. Specifies filename for simulation output.
`IOmode`	A string that specifies the input-output mode for the time series = "verbose", "dev" or "suppress".

Details

modelTag provides the main function with requested models. modelTag is vector of any of the following supported models:

- `"Simple-ann"`a simple annual scaling
- `"P-ann-wgen"`a four parameter annual rainfall model
- `"P-seas-wgen"`a 16 parameter seasonal rainfall model
- `"P-har6-wgen"`a harmonic rainfall model with 6 periods
- `"P-har12-wgen"`a harmonic rainfall model
- `"P-har12-wgen-FS"`a harmonic rainfall model where seasonality is fixed (phase angles must be specified via modelInfoMod=list("P-har12-wgen-FS"=fixedPars=c(x,x,x,x))
- `"P-har26-wgen"`a harmonic rainfall model
- `"P-2har26-wgen"`a double harmonic rainfall model
- `"Temp-har26-wgen"`a harmonic temperature model not conditional on rainfall
- `"Temp-har26-wgen-wd"`a harmonic temperature model dependent on wet or dry day
- `"Temp-har26-wgen-wdsd"`a harmonic temperature model where standard deviation parameters are dependent on wet or dry day
- `"PET-har12-wgen"`a harmonic potential evapotranspiration model
- `"PET-har26-wgen"`a harmonic potential evapotranspiration model
- `"PET-har26-wgen-wd"`a harmonic potential evapotranspiration model dependent on wet or dry day
- `"Radn-har26-wgen"`a harmonic solar radiation model (MJ/m2)

Examples

```
## Not run:
data(tankDat); obs=tank_obs                       #Get observed data
modelTag=c("P-har12-wgen","Temp-har26-wgen")      #Select models
pars=modCalibrator(obs=obs,modelTag=modelTag)     #Calibrate models
sim=modSimulator(datStart="1970-01-01",           #Simulate!
                 datFinish="1999-12-31",
                 modelTag=modelTag,
                 parS=pars,
                 seed=123,
                 file=paste0("tester.csv"),
                 IOmode="verbose")
plot(sim$P[1:365])                               #Plot first year of rainfall

## End(Not run)
```

`oat_out` *One-at-a-time testing scenarios for use in tank model example*

Description

A list of example one-at-a-time testing scaling scenarios

`performanceSpaces` *performanceSpaces*

Description

Simulates and maps system performance using climate scenarios generated using `scenarioGenerator`. Currently only visualises 2D panels.

Usage

```
performanceSpaces(data = NULL,
                  plotTag = NULL,
                  plotArgs = NULL,
                  systemModel = NULL,
                  systemArgs = NULL,
                  simDirectory = "Simulation1",
                  performance=NULL,
                  IOmode="suppress"
                  )
```

Arguments

`data` A list containing data frames of the form *Year Month Day P Temp* output from `scenarioGenerator`

`plotTag` A character vector which designates how the performance space is visualised. Options are: "Heat", "Contours" or "OAT".

`plotArgs` A list controlling how the performance space is visualised with the following components:

- **title** a string that is used as the title label.
- **legendtitle** a string that is used to label the legend.
- **xlim** x axis limits.
- **ylim** y axis limits.
- **performancelimits** z axis limits.
- **lowfill** a string specifying the colour for lower values of performance. Default is "red". For use with plotTag="Heat".
- **highfill** a string specificying the colour for the high values of performance. Default is "yellow". A colour ramp will be created between this and the lowfill. For use with plotTag="Heat".
- **contour** a TRUE/FALSE toggle for whether or not contour lines are overlaid. Default is TRUE. For use with plotTag="Heat".
- **contourlevels** a vector specifying the levels at which controus will be drawn. For use with plotTag = "Contours".

`systemModel` a function name. The function name of the system model used to generate system performance. The system model must have the arguments *data* and *systemArgs*.

`systemArgs` A list containing all the arguments that are required to control the system model.

`simDirectory` A string used to label the output directory.

`performance` A vector of performance values from an external system model.

`IOmode` A string that specifies the input-output mode for the scenarios = "verbose", "dev" or "suppress".

Value

Returns a 2D plot of system performance.

See Also

See Also: `scenarioGenerator` and `plotLayers`

Examples

```
## Not run:
data(tankSimpleScale)

# tank_simple_scenarios<-scenarioGenerator(obs=tank_obs,
#                            modelTag = modelTag,
```

```
#                           attPerturb=attPerturb,
#                           exSpArgs = exSpArgs)

# Example 1: Heat Map Example
systemArgs<-list(roofArea=100,
                 nPeople=1,
                 tankVol=2000,
                 firstFlush=1,
                 write.file=FALSE,
                 metric="reliability")

plotArgs=list(title="Scenario Neutral Space",
               legendtitle="Reliability",
               xlim=c(-2,2),
               ylim=c(0.7,1.3),
               performancelimits=c(0.6,0.85))

plot<-performanceSpaces(data=tank_simple_scenarios,
                        plotTag="Heat",
                        systemModel = tankWrapper,
                        systemArgs = systemArgs)

plot$plot

# Note options
# plotArgs$contour=FALSE
# plotArgs$lowfill="antiquewhite"   ###From supported R colour names
# plotArgs$highfill="#88CCEE"       ###Hexidecimal specification also okay

## End(Not run)

## Not run:
#Example 2
result<-performanceSpaces(data=tank_simple_scenarios,
                          plotTag = "Contours",
                          plotArgs=plotArgs,
                          systemModel = tankWrapper,
                          systemArgs = systemArgs)

result$plot

## End(Not run)
#Example 3
## Not run:
plotArgs$contourlevels=c(0.67,0.71)

result<-performanceSpaces(data=tank_simple_scenarios,
                          plotTag = "Contours",
                          plotArgs=plotArgs,
                          systemModel = tankWrapper,
                          systemArgs = systemArgs)

result$plot
```

```
## End(Not run)

## Not run:
###Example 4 - One-at-a-time/"OAT" plot
data(oatScenarios)
systemArgs=list(roofArea=50,
                 nPeople=1,
                 tankVol=3000,
                 firstFlush=1,
                 write.file=FALSE,
                 metric="reliability")

plotArgs=list(title="Scenario Neutral Space",
               legendtitle="Reliability",
               xlim=c(-2,2),
               ylim=c(0.7,1.3),
               performancelimits=c(0.6,0.85))

oat_plot=performanceSpaces(data=oat_out,
                           plotTag = "OAT",
                           plotArgs=plotArgs,
                           systemModel = tankWrapper,
                           systemArgs = systemArgs)
oat_plot

## End(Not run)
```

---

plotLayers *plotLayers*

Description

Superimposes climate projection information on to the 2D system performance spaces generated using performanceSpaces

Usage

```
plotLayers(plot = NULL,
           plotArgs = NULL,
           climdata = NULL,
           climArgs = list(),
           simDirectory = "Simulation1",
           IOmode="suppress"
                )
```

Arguments

`plot` A, editable plot output from `performanceSpaces`. The editable plot can be accessed via plot$plotEdit.

`plotArgs` A list controlling how the performance space is visualised with the following components:

- **title** a string that is used as the title label.
- **legendtitle** a string that is used to label the legend.
- **xlim** x axis limits.
- **ylim** y axis limits.
- **performancelimits** z axis limits.
- **lowfill** a string specifying the colour for lower values of performance. Default is 'red". For use with plotTag="Heat".
- **highfill** a string specificying the colour for the high values of performance. Default is "yellow". A colour ramp will be created between this and the lowfill. For use with plotTag="Heat".
- **contour** a TRUE/FALSE toggle for whether or not contour lines are overlaid. Default is TRUE. For use with plotTag="Heat".
- **contourlevels** a vector specifying the levels at which controus will be drawn. For use with plotTag = "Contours".

`climdata` a data frame of the projected climate data

`climArgs` a list that controls the appearance of the superimposed climate projections with the following components:

- **colour** a string that is used to set the colour of the superimposed climate drojections. Defaults to black.
- **fill** a string that is used to determine whether the system performance resulting from the climate projections should also be displayed. Set to "performance" to infill climate projections according to their system performance.

`simDirectory` A string used to label the output directory.

`IOmode` A string that specifies the input-output mode for the scenarios = "verbose", "dev" or "suppress".

Value

climate projection information superimposed on a 2D system performance spaces plot.

See Also

See Also: `scenarioGenerator`, `performanceSpaces` and `quickSpace`

Examples

```
## Not run:
data(tankPlot)
data(climdata2030)          #loading climate data for 2030 time slice
```

```
###Example 1
# Saved from a previous example
# tank_simpleScale_plot<-performanceSpaces(data=tank_simple_scenarios,
#                                          plotTag = "Heat",
#                                          plotArgs=plotArgs,
#                                          systemModel = tankWrapper,
#                                          systemArgs = systemArgs)

#Create plotting arguments
plotArgs<-list(title="Scenario neutral space with projections overlaid",
               ylim=c(0.7,1.3),
               xlim=c(-2,2),
               xtitle="Temp_ann_avg_m",
               ytitle="P_ann_tot_m")

climArgs<-list(performancelimits=NULL,
               label=NULL,
               slice=2030,
               colour="black",
               fill="performance")

#Plot performance space with projections overlaid
tank_overlay_plot=plotLayers(plot=tank_simpleScale_plot$plotEdit,
                             plotArgs=plotArgs,
                             climdata=climdata,
                             climArgs=climArgs)
tank_overlay_plot

## End(Not run)
```

`quickSpace` *quickSpace*

Description

Master function - generates perturbed climate scenarios, generates performance, visualizes performanace and superimposes climate projection information on to the system performance spaces

Usage

```
quickSpace(obs=NULL,
           modelTag="Simple-ann",
           attPerturb=NULL,
           attHold=NULL,
           attPenalty=NULL,
           optimArgs=list(pcrossover= 0.8,
                          pmutation=0.1,
                          maxiter=10,
                          maxFitness=-0.001,
```

```
                    popSize = 100,
                    run=20,
                    seed = NULL,
                    parallel = FALSE,
                    keepBest=TRUE,
                    lambda.mult=0.0
      ),
  exSpArgs=list(),
  simLengthNyrs=NULL,
  systemModel=NULL,
  systemArgs=NULL,
  plotTag="Heat",
  plotArgs=NULL,
  IOmode="suppress",
  arrayID=NULL,
  nSeed=NULL,
  climdata=NULL,
  climArgs=list(),
  simDirectory="Simulation1")
```

Arguments

`obs` A dataframe of observed climate data in the form *Year Month Day P Temp*.

`modelTag` A character vector of which stochastic models to use to create each climate variable.Supported tags are shown under details below.

`attPerturb` A character vector of climate attributes to hold at a target. A list of all supported attributes can be found under details below.

`attHold` A character vector of climate attributes to hold at a target. A list of all supported attributes can be found under under details below.

`attPenalty` A character vector of climate attributes to place specific focus on during targeting via the use of a penalty function during the optimisation process.

`optimArgs` A list controlling the search algorithm with the following components:

- `pcrossover` a value of probability of crossover. Defaults to 0.8.
- `pmutation` a value of probability of mutation. Defaults to 0.1
- `maxiter` a value of the maximum number of generations. Defaults to 50
- `maxFitness` a value of the stopping criteria. Defaults to 0.001
- `popSize` a value of the population size. Defaults to 100
- `run` a value of an alternative stopping criteria - consecutive runs without improvement in fitness. Defaults to 0.001
- `seed` a value of the random seed. Defaults to NULL
- `parallel` specifies if parallel computing should be used. Defaults to False. Can be set to the number of desired cores, or `TRUE`, where it will detect the number of available cores and run.
- `keepBest` specifies if the optimisation should keep the best solution in each generation. Defaults to TRUE.

- `lambda.mult` A multiplier used during the optimisation for primary attributes. Defaults to zero.

`exSpArgs` a list to control the exposure space creation with the following components:

- `type` a string that specifies the type of sampling. Defaults to regular spacing.
- `samp` a vector indicating the number of targets for each attribute in attSel.
- `bounds` a list containing elements for attributes listed in attSel, where each attribute has bounds specified. This should be a single value for a stationary target, and a vector of min and max change for primary attributes. Works with samp to create number of step sizes. Defaults with samp to only reproduce historical weather.

`simLengthNyrs` A scalar that specifies the length in years of each generated scenario. Only used with stochastic generation.

`systemModel` a function name. The function name of the system model used to generate system performance. The system model must have the arguments *data* and *systemArgs*.

`systemArgs` A list containing all the arguments that are required to control the system model.

`plotTag` A character vector which designates how the performance space is visualised. Options are: "Heat", "Contours" or "OAT".

`plotArgs` A list controlling how the performance space is visualised with the following components:

- **title** a string that is used as the title label.
- **legendtitle** a string that is used to label the legend.
- **xlim** x axis limits.
- **ylim** y axis limits.
- **performancelimits** z axis limits.
- **lowfill** a string specifying the colour for lower values of performance. Default is "red". For use with plotTag="Heat".
- **highfill** a string specificying the colour for the high values of performance. Default is "yellow". A colour ramp will be created between this and the lowfill. For use with plotTag="Heat".
- **contour** a TRUE/FALSE toggle for whether or not contour lines are overlaid. Default is TRUE. For use with plotTag="Heat".
- **contourlevels** a vector specifying the levels at which controus will be drawn. For use with plotTag = "Contours".

`IOmode` A string that specifies the input-output mode for the scenarios = "verbose", "dev" or "suppress".

`arrayID` A scalar used to append output file names - only used in IOmode="dev".

`nSeed` A scalar used to specify the number of seeds (in this case replicates) for the stochastic generation of time series.

`climdata` a data frame of the projected climate data

`climArgs` a list that controls the appearance of the superimposed climate projections with the following components:

colour a string that is used to set the colour of the superimposed climate drojections. Defaults to black.

fill a string that is used to determine whether the system performance resulting from the climate projections should also be displayed. Set to "Performance" to infill climate projections according to their system performance.

simDirectory A string used to label the output directory.

Details

modelTag provides the main function with requested models. modelTag is vector of any of the following supported models:

1. "Simple-ann"a simple annual scaling
2. "P-ann-wgen"a four parameter annual rainfall model
3. "P-seas-wgen"a 16 parameter seasonal rainfall model
4. "P-har6-wgen"a harmonic rainfall model with 6 periods
5. "P-har12-wgen"a harmonic rainfall model
6. "P-har12-wgen-FS"a harmonic rainfall model where seasonality is fixed (phase angles must be specified via modelInfoMod=list("P-har12-wgen-FS"=fixedPars=c(x,x,x,x))
7. "P-har26-wgen"a harmonic rainfall model
8. "P-2har26-wgen"a double harmonic rainfall model
9. "Temp-har26-wgen"a harmonic temperature model not conditional on rainfall
10. "Temp-har26-wgen-wd"a harmonic temperature model dependent on wet or dry day
11. "Temp-har26-wgen-wdsd"a harmonic temperature model where standard deviation parameters are dependent on wet or dry day
12. "PET-har12-wgen"a harmonic potential evapotranspiration model
13. "PET-har26-wgen"a harmonic potential evapotranspiration model
14. "PET-har26-wgen-wd"a harmonic potential evapotranspiration model dependent on wet or dry day
15. "Radn-har26-wgen"a harmonic solar radiation model (MJ/m2)

The list of attributes supported by attSel are:

1. "P_ann_tot_m"
2. "P_ann_R10_m"
3. "P_ann_maxDSD_m"
4. "P_ann_maxWSD_m"
5. "P_ann_P99_m"
6. "P_ann_dyWet99p_m"
7. "P_ann_ratioWS_m"
8. "Temp_ann_avg_m"
9. "Temp_ann_P5_m"

10. "Temp_ann_P95_m"
11. "Temp_ann_F0_m"
12. "P_ann_dyWet_m"
13. "P_ann_DSD_m"
14. "P_seas_tot_cv"
15. "P_mon_tot_cv"
16. "P_ann_avgWSD_m"
17. "P_ann_avgDSD_m"
18. "P_JJA_avgWSD_m"
19. "P_MAM_avgWSD_m"
20. "P_DJF_avgWSD_m"
21. "P_SON_avgWSD_m"
22. "P_JJA_avgDSD_m"
23. "P_MAM_avgDSD_m"
24. "P_DJF_avgDSD_m"
25. "P_SON_avgDSD_m"
26. "Temp_ann_GSL_m"
27. "Temp_ann_CSL_m"
28. "P_JJA_dyWet_m"
29. "P_MAM_dyWet_m"
30. "P_DJF_dyWet_m"
31. "P_SON_dyWet_m"
32. "P_JJA_tot_m"
33. "P_MAM_tot_m"
34. "P_DJF_tot_m"
35. "P_SON_tot_m"
36. "P_ann_nWet_m"
37. "P_ann_dyAll_m"
38. "P_JJA_dyAll_m"
39. "P_MAM_dyAll_m"
40. "P_DJF_dyAll_m"
41. "P_SON_dyAll_m"
42. "PET_ann_avg_m"
43. "PET_ann_tot_m"
44. "PET_ann_rng_m"
45. "Temp_ann_rng_m"
46. "PET_ann_90pX_m"

47. "P_ann_90X_m"
48. "P_ann_seasRatio_m"
49. "PET_ann_P5_m"
50. "PET_ann_P95_m"
51. "P_Jan_tot_m"
52. "P_Feb_tot_m"
53. "P_Mar_tot_m"
54. "P_Apr_tot_m"
55. "P_May_tot_m"
56. "P_Jun_tot_m"
57. "P_Jul_tot_m"
58. "P_Aug_tot_m"
59. "P_Sep_tot_m"
60. "P_Oct_tot_m"
61. "P_Nov_tot_m"
62. "P_Dec_tot_m"
63. "Temp_JJA_avg_m"
64. "Temp_MAM_avg_m"
65. "Temp_DJF_avg_m"
66. "Temp_SON_avg_m"
67. "Temp_Jan_avg_m"
68. "Temp_Feb_avg_m"
69. "Temp_Mar_avg_m"
70. "Temp_Apr_avg_m"
71. "Temp_May_avg_m"
72. "Temp_Jun_avg_m"
73. "Temp_Jul_avg_m"
74. "Temp_Aug_avg_m"
75. "Temp_Sep_avg_m"
76. "Temp_Oct_avg_m"
77. "Temp_Nov_avg_m"
78. "Temp_Dec_avg_m"
79. "PET_JJA_avg_m"
80. "PET_MAM_avg_m"
81. "PET_DJF_avg_m"
82. "PET_SON_avg_m"
83. "PET_JJA_tot_m"

84. "PET_MAM_tot_m"
85. "PET_DJF_tot_m"
86. "PET_SON_tot_m"
87. "PET_Jan_tot_m"
88. "PET_Feb_tot_m"
89. "PET_Mar_tot_m"
90. "PET_Apr_tot_m"
91. "PET_May_tot_m"
92. "PET_Jun_tot_m"
93. "PET_Jul_tot_m"
94. "PET_Aug_tot_m"
95. "PET_Sep_tot_m"
96. "PET_Oct_tot_m"
97. "PET_Nov_tot_m"
98. "PET_Dec_tot_m"
99. "PET_Jan_avg_m"
100. "PET_Feb_avg_m"
101. "PET_Mar_avg_m"
102. "PET_Apr_avg_m"
103. "PET_May_avg_m"
104. "PET_Jun_avg_m"
105. "PET_Jul_avg_m"
106. "PET_Aug_avg_m"
107. "PET_Sep_avg_m"
108. "PET_Oct_avg_m"
109. "PET_Nov_avg_m"
110. "PET_Dec_avg_m"
111. "PET_ann_seasRatio_m"
112. "Radn_ann_avg_m"
113. "Radn_ann_tot_m"
114. "Radn_ann_rng_m"
115. "Radn_ann_P5_m"
116. "Radn_ann_P95_m"
117. "Radn_JJA_avg_m"
118. "Radn_MAM_avg_m"
119. "Radn_DJF_avg_m"
120. "Radn_SON_avg_m"

121. "Radn_JJA_tot_m"
122. "Radn_MAM_tot_m"
123. "Radn_DJF_tot_m"
124. "Radn_SON_tot_m"
125. "Radn_Jan_tot_m"
126. "Radn_Feb_tot_m"
127. "Radn_Mar_tot_m"
128. "Radn_Apr_tot_m"
129. "Radn_May_tot_m"
130. "Radn_Jun_tot_m"
131. "Radn_Jul_tot_m"
132. "Radn_Aug_tot_m"
133. "Radn_Sep_tot_m"
134. "Radn_Oct_tot_m"
135. "Radn_Nov_tot_m"
136. "Radn_Dec_tot_m"
137. "Radn_Jan_avg_m"
138. "Radn_Feb_avg_m"
139. "Radn_Mar_avg_m"
140. "Radn_Apr_avg_m"
141. "Radn_May_avg_m"
142. "Radn_Jun_avg_m"
143. "Radn_Jul_avg_m"
144. "Radn_Aug_avg_m"
145. "Radn_Sep_avg_m"
146. "Radn_Oct_avg_m"
147. "Radn_Nov_avg_m"
148. "Radn_Dec_avg_m"
149. "Radn_ann_seasRatio_m"

scenarioGenerator *Scenario Generator*

Description

Produces the data needed for an exposure space of requested climate variables, attributes and bounds.

Usage

```
scenarioGenerator(obs = NULL,
                  modelTag = NULL,
                  modelInfoMod=list(),
                  attPerturb = NULL,
                  attHold=NULL,
                  attPenalty = NULL,
                  optimArgs = list(pcrossover= 0.8,
                                   pmutation=0.1,
                                   maxiter=10,
                                   maxFitness=-0.001,
                                   popSize = 100,
                                   run=20,
                                   seed = 1234,
                                   parallel = FALSE,
                                   keepBest=TRUE,
                                   lambda.mult = 0,
                                   suggestions=NULL
                                   ),
                  exSpArgs=list(),
                  simLengthNyrs=NULL,
                  IOmode="suppress",
                  arrayID=NULL,
                  nSeed=NULL,
                  seedID=1234,
                  simDirectory="Simulation1"
                  )
```

Arguments

obs	A dataframe of observed climate data in the form *Year Month Day P Temp*.
modelTag	A character vector of which stochastic models to use to create each climate variable.Supported tags are shown under details below.
modelInfoMod	A list containing information for modifying stochastic model bounds and defining fixed parameters.
attPerturb	A character vector of climate attributes to hold at a target. A list of all supported attributes can be found under shown under details below.

`attHold` A character vector of climate attributes to hold at a target. A list of all supported attributes can be found under shown under details below.

`attPenalty` A character vector of climate attributes to place specific focus on during targeting via the use of a penalty function during the optimisation process.

`optimArgs` A list controlling the search algorithm with the following components:

- `pcrossover` a value of probability of crossover. Defaults to 0.8.
- `pmutation` a value of probability of mutation. Defaults to 0.1
- `maxiter` a value of the maximum number of generations. Defaults to 50
- `maxFitness` a value of the stopping criteria. Defaults to 0.001
- `popSize` a value of the population size. Defaults to 100
- `run` a value of an alternative stopping criteria - consecutive runs without improvement in fitness. Defaults to 0.001
- `seed` a value of the random seed. Defaults to 1234.
- `parallel` specifies if parallel computing should be used. Defaults to False. Can be set to the number of desired cores, or `TRUE`, where it will detect the number of available cores and run.
- `keepBest` specifies if the optimisation should keep the best solution in each generation. Defaults to TRUE.
- `lambda.mult` A multiplier used during the optimisation for primary attributes. Defaults to zero.

`exSpArgs` a list to control the exposure space creation with the following components:

- `type` a string that specifies the type of sampling. Defaults to regular spacing.
- `samp` a vector indicating the number of targets for each attribute in attSel.
- `bounds` a list containing elements for attributes listed in attSel, where each attribute has bounds specified. This should be a single value for a stationary target, and a vector of min and max change for primary attributes. Works with samp to create number of step sizes. Defaults with samp to only reproduce historical weather.

`simLengthNyrs` A scalar that specifies the length in years of each generated scenario. Only used with stochastic generation.

`IOmode` A string that specifies the input-output mode for the scenarios = "verbose", "dev" or "suppress".

`arrayID` A scalar used to append output file names - only used in IOmode="dev".

`nSeed` A scalar used to specify the number of seeds (in this case replicates) for the stochastic generation of time series.

`seedID` determines how seeds are used - option "fixed","arrayID" or a number to set the seed.

`simDirectory` A string used to label the output directory.

Details

modelTag provides the main function with requested models. modelTag is vector of any of the following supported models:

- "Simple-ann"a simple annual scaling
- "P-ann-wgen"a four parameter annual rainfall model
- "P-seas-wgen"a 16 parameter seasonal rainfall model
- "P-har6-wgen"a harmonic rainfall model with 6 periods
- "P-har12-wgen"a harmonic rainfall model
- "P-har12-wgen-FS"a harmonic rainfall model where seasonality is fixed (phase angles must be specified via modelInfoMod=list("P-har12-wgen-FS"=fixedPars=c(x,x,x,x))
- "P-har26-wgen"a harmonic rainfall model
- "P-2har26-wgen"a double harmonic rainfall model
- "Temp-har26-wgen"a harmonic temperature model not conditional on rainfall
- "Temp-har26-wgen-wd"a harmonic temperature model dependent on wet or dry day
- "Temp-har26-wgen-wdsd"a harmonic temperature model where standard deviation parameters are dependent on wet or dry day
- "PET-har12-wgen"a harmonic potential evapotranspiration model
- "PET-har26-wgen"a harmonic potential evapotranspiration model
- "PET-har26-wgen-wd"a harmonic potential evapotranspiration model dependent on wet or dry day
- "Radn-har26-wgen"a harmonic solar radiation model (MJ/m2)

The list of attributes supported by attSel are:

- "P_ann_tot_m"
- "P_ann_R10_m"
- "P_ann_maxDSD_m"
- "P_ann_maxWSD_m"
- "P_ann_P99_m"
- "P_ann_dyWet99p_m"
- "P_ann_ratioWS_m"
- "Temp_ann_avg_m"
- "Temp_ann_P5_m"
- "Temp_ann_P95_m"
- "Temp_ann_F0_m"
- "P_ann_dyWet_m"
- "P_ann_DSD_m"
- "P_seas_tot_cv"
- "P_mon_tot_cv"
- "P_ann_avgWSD_m"
- "P_ann_avgDSD_m"
- "P_JJA_avgWSD_m"

- "P_MAM_avgWSD_m"
- "P_DJF_avgWSD_m"
- "P_SON_avgWSD_m"
- "P_JJA_avgDSD_m"
- "P_MAM_avgDSD_m"
- "P_DJF_avgDSD_m"
- "P_SON_avgDSD_m"
- "Temp_ann_GSL_m"
- "Temp_ann_CSL_m"
- "P_JJA_dyWet_m"
- "P_MAM_dyWet_m"
- "P_DJF_dyWet_m"
- "P_SON_dyWet_m"
- "P_JJA_tot_m"
- "P_MAM_tot_m"
- "P_DJF_tot_m"
- "P_SON_tot_m"
- "P_ann_nWet_m"
- "P_ann_dyAll_m"
- "P_JJA_dyAll_m"
- "P_MAM_dyAll_m"
- "P_DJF_dyAll_m"
- "P_SON_dyAll_m"
- "PET_ann_avg_m"
- "PET_ann_tot_m"
- "PET_ann_rng_m"
- "Temp_ann_rng_m"
- "PET_ann_90pX_m"
- "P_ann_90X_m"
- "P_ann_seasRatio_m"
- "PET_ann_P5_m"
- "PET_ann_P95_m"
- "P_Jan_tot_m"
- "P_Feb_tot_m"
- "P_Mar_tot_m"
- "P_Apr_tot_m"
- "P_May_tot_m"

- "P_Jun_tot_m"
- "P_Jul_tot_m"
- "P_Aug_tot_m"
- "P_Sep_tot_m"
- "P_Oct_tot_m"
- "P_Nov_tot_m"
- "P_Dec_tot_m"
- "Temp_JJA_avg_m"
- "Temp_MAM_avg_m"
- "Temp_DJF_avg_m"
- "Temp_SON_avg_m"
- "Temp_Jan_avg_m"
- "Temp_Feb_avg_m"
- "Temp_Mar_avg_m"
- "Temp_Apr_avg_m"
- "Temp_May_avg_m"
- "Temp_Jun_avg_m"
- "Temp_Jul_avg_m"
- "Temp_Aug_avg_m"
- "Temp_Sep_avg_m"
- "Temp_Oct_avg_m"
- "Temp_Nov_avg_m"
- "Temp_Dec_avg_m"
- "PET_JJA_avg_m"
- "PET_MAM_avg_m"
- "PET_DJF_avg_m"
- "PET_SON_avg_m"
- "PET_JJA_tot_m"
- "PET_MAM_tot_m"
- "PET_DJF_tot_m"
- "PET_SON_tot_m"
- "PET_Jan_tot_m"
- "PET_Feb_tot_m"
- "PET_Mar_tot_m"
- "PET_Apr_tot_m"
- "PET_May_tot_m"
- "PET_Jun_tot_m"

- "PET_Jul_tot_m"
- "PET_Aug_tot_m"
- "PET_Sep_tot_m"
- "PET_Oct_tot_m"
- "PET_Nov_tot_m"
- "PET_Dec_tot_m"
- "PET_Jan_avg_m"
- "PET_Feb_avg_m"
- "PET_Mar_avg_m"
- "PET_Apr_avg_m"
- "PET_May_avg_m"
- "PET_Jun_avg_m"
- "PET_Jul_avg_m"
- "PET_Aug_avg_m"
- "PET_Sep_avg_m"
- "PET_Oct_avg_m"
- "PET_Nov_avg_m"
- "PET_Dec_avg_m"
- "PET_ann_seasRatio_m"
- "Radn_ann_avg_m"
- "Radn_ann_tot_m"
- "Radn_ann_rng_m"
- "Radn_ann_P5_m"
- "Radn_ann_P95_m"
- "Radn_JJA_avg_m"
- "Radn_MAM_avg_m"
- "Radn_DJF_avg_m"
- "Radn_SON_avg_m"
- "Radn_JJA_tot_m"
- "Radn_MAM_tot_m"
- "Radn_DJF_tot_m"
- "Radn_SON_tot_m"
- "Radn_Jan_tot_m"
- "Radn_Feb_tot_m"
- "Radn_Mar_tot_m"
- "Radn_Apr_tot_m"
- "Radn_May_tot_m"

- "Radn_Jun_tot_m"
- "Radn_Jul_tot_m"
- "Radn_Aug_tot_m"
- "Radn_Sep_tot_m"
- "Radn_Oct_tot_m"
- "Radn_Nov_tot_m"
- "Radn_Dec_tot_m"
- "Radn_Jan_avg_m"
- "Radn_Feb_avg_m"
- "Radn_Mar_avg_m"
- "Radn_Apr_avg_m"
- "Radn_May_avg_m"
- "Radn_Jun_avg_m"
- "Radn_Jul_avg_m"
- "Radn_Aug_avg_m"
- "Radn_Sep_avg_m"
- "Radn_Oct_avg_m"
- "Radn_Nov_avg_m"
- "Radn_Dec_avg_m"
- "Radn_ann_seasRatio_m"

Examples

```
library(foreSIGHT)                 ###Load package

data(tankDat)                         ###Load tank data

###Scenario generation arguments
modelTag="Simple-ann"
attPerturb<-c("P_ann_tot_m","Temp_ann_avg_m")
exSpArgs<-list(type = "regGrid",
               samp = c(7,6),
               bounds = list("P_ann_tot_m"=c(0.9,1.5),
                             "Temp_ann_avg_m"=c(-1,4)))

###Function call
out<-scenarioGenerator(obs=tank_obs,
                  modelTag = modelTag,
                  attPerturb=attPerturb,
                  exSpArgs = exSpArgs,
                  simDirectory="Simulation1")
```

tankPerformance — *A function to calculate difference performance from simulated tank behaviour*

Description

A function to calculate difference performance from simulated tank behaviour

Usage

```
tankPerformance(data=NULL,
                      roofArea=50,
                      nPeople=1,
                      tankVol=3000,
      firstFlush=1,
                      write.file=TRUE,
                      fnam="tankperformance.csv")
```

Arguments

data	A dataframe of observed climate data in the form *Year Month Day P Temp*.
roofArea	roof area in m2
nPeople	number of people using water
tankVol	tank volume in L
firstFlush	first flush depth over roof in mm
write.file	logical. write output tank timeseries to file T/F?
fnam	string indicating name of file

tankWrapper — *A demo tank model of the format required for uses in foreSIGHT examples and vignette*

Description

A demo tank model with wrapper to tailor format to foreSIGHT

Usage

```
tankWrapper(data=NULL,
            systemArgs=NULL,
            repID=NULL)
```

Arguments

data
: A dataframe of observed climate data in the form *Year Month Day P Temp*.

systemArgs
: a list to control the exposure space creation with the following components:
  - `roofArea` roof area in m2
  - `nPeople` number of people using water
  - `tankVol` tank volume in L
  - `firstFlush` first flush depth over roof in mm
  - `write.file` logical. write output tank timeseries to file T/F?
  - `fnam` string indicating name of file
  - `metric` string indicating what metric to report: average daily deficit - "avDeficit", volumetric reliability - "volRel", reliability - "reliability", storage efficiency - "storEff", system efficiency - "sysEff"

repID
: a number that is used of append file names to distiguish between replicates.

---

tank_obs — *Observations for demo tank model examples and vignette*

Description

Dataset of observations for tank model examples

Format

A dataframe of observed climate data in the form *Year Month Day P Temp*.

---

tank_simpleScale_plot — *Example plot of tank model stress tested using simple scaling*

Description

A list of plots displaying the tank model stress tested using simple scaling for use in example of `plotLayers`.

Format

tank_simpleScale_plot is a list of plots containing

- `plot` static plot of performance
- `plotEdit` editable plot of performance for use in `plotLayers`.

---

tank_simple_scenarios — *Simple scaling scenarios for use in tank model example*

Description

A list of example simple scaling scenarios

Index