Package 'daRt'

August 21, 2019

Type Package

Title Read DART Model Outputs

Version 0.5.0

Author William T. J. Morrison

Maintainer William T. J. Morrison <willmorrison661@gmail.com>

Description For reading outputs from the Discrete Anisotropic Radiative Transfer (DART) model, formatted in a ``long'' dplyr-ready format suitable for efficient analysis.

Github https://github.com/willmorrison1/daRt

License GPL-3

Encoding UTF-8

RoxygenNote 6.1.1

R topics documented:

accessors *Access object information*

Description

Generic functions to access information from the objects with classes defined in this package

Usage

```
product(x)

simname(x)

files(x)

bands(x)

iters(x)

variables(x)

variablesRB3D(x)

typeNums(x)

imageType(x)

imageNo(x)
```

as.data.frame,SimulationData-method *as.data.frame*

Description

as.data.frame

Usage

```
## S4 method for signature 'SimulationData'
as.data.frame(x, as.tibble = TRUE)
```

Arguments

`x`	SimulationData.

`Directions-class` *Directions data class*

Description

Directions data class that extends SimulationData-class class.

`getData` *Main function: get DART data*

Description

Main function to get data from DART simulation outputs in a friendly 'long' data format that is part of an object that extends a SimulationData-class type object

Usage

```
getData(x, sF, ...)
```

Arguments

`x` simulation directory or directories (character) or SimulationFiles-class object

`sF` SimulationFilter-class if x = `character`

`getFiles` *Get DART output filenames*

Description

Get DART output filenames

Usage

```
getFiles(x = "character", sF = "SimulationFilter")
```

Arguments

`x` simulation directory or directories (character)

`sF` SimulationFilter-class object

`...` Optional arguments of: `nCores`: number of cores to use when loading data.

`Images-class` *Images data class*

Description

Image data class extends SimulationData-class class.

imagesToDirectionsDF *imagesToDirectionsDF*

Description

Convert an Images-class object to a Directions-class object

Usage

```
imagesToDirectionsDF(x, fun)
```

Arguments

`x`	Images-class object
`fun`	Function to apply across each image.

Details

Aggregate images to single values

plotDirections *plotDirections*

Description

Plot directions data as polar plot.

Usage

```
plotDirections(azimuth, zenith, value, azimuthOffsetVal = 0,
  outerRadius = max(zenith) + max(zenith) * 0.01, zenithLabPch = 20,
  zenithLabCol = "darkgrey", zenithLabCex = 1, brks = seq(min(value),
  max(value), length.out = 10), cols = c("dark grey",
  colorRampPalette(c("purple", "blue3", "yellow", "red"))(length(brks) -
  3), "firebrick4"), ...)
```

Arguments

`azimuth`	Numeric. Azimuth angle with DART conventions
`zenith`	Numeric. Zenith angle with DART conventions
`value`	Numeric. Values associated with the given azimuth and zenith angles
`azimuthOffsetVal`	Numeric. Scene offset (degrees) as shown in the DART GUI.
`outerRadius`	Numeric. Maximum radius (degrees) of polar plot
`zenithLabPch`	Numeric. Pch for zenith label.
`zenithLabCol`	Character. Colour for zenith label.
`zenithLabCex`	Numeric. Cex for zenith label.
`brks`	Numeric. Breaks for colour palette e.g. seq(0, 1, by = 0.1). Optional.
`cols`	Character. Colours for given breaks. Optional.
`...`	Additional options passed to points() when drawing directions points.

Examples

```
#Inputs are DART oriented directions (as seen in the DART files and \link{Directions-class})
plotDirections(azimuth = rep(225, 10),
               zenith = seq(0, 90, length.out = 10),
               value = 1:10)
#Output plot uses 'upward' directions from ground, where e.g.:
  0deg (270deg) azimuth faces north (west)
  0deg (90deg) zenith faces upward (horizon)
```

`RB3D-class` *RB3D class*

Description

RB3D (Radiative Budget 3D) class that extends SimulationData-class class.

`SimulationData-class` *Generic SimulationData class*

Description

Generic SimulationData class that extends to data classes for specific DART products

Slots

`data` data.frame.

See Also

Images-class Directions-class RB3D-class

`SimulationFiles-class` *SimulationFiles class*

Description

An S4 class to represent the files within a simulation or simulations. Created using the `getFiles` method. Specific files within the class are modified by the object with class SimulationFilter-class

Usage

```
simdir(x)
```

Slots

`simulationFilter` contains SimulationFilter-class object

`files` a data.frame, with each row describing the file

`sequenceInfoList` a list, with each list element showing the variable permutation(s) within this specific simulation sequence.

simulationFilter *Create SimulationFilter class*

Description

Function for creating the SimulationFilter class

Usage

```
simulationFilter(product = "character", ...)
```

Arguments

product	One of "directions", "rb3D", "images".
...	

See Also

SimulationFilter-class

SimulationFilter-class *SimulationFilter class.*

Description

SimulationFilter class.

Usage

```
product(x) <- value

iters(x) <- value

bands(x) <- value

variablesRB3D(x) <- value

variables(x) <- value

typeNums(x) <- value

imageType(x) <- value

imageNo(x) <- value
```

Slots

- `bands` character.
- `variables` character.
- `iters` character.
- `variablesRB3D` character.
- `typeNums` character.
- `imageType` character.
- `imageNo` numeric.
- `product` character.

See Also

`simulationFilter`

Index