Wikiprint Book

Title: dataInventory

Subject: TracMeteo - EcomsUdg/RPackage/Functions

Version: 44

Date: 10/23/2021 07:59:13 PM

Table of Contents

dataInventory

Description

Provides summary information about the main characteristics of a NcML dataset.

Usage

```
dataInventory(ncml.file)
```

Arguments

- `ncml.file`: a character string indicating the full path to the virtual dataset (the `NcML` file). This can be either a path containing the directory and name of the file, or an appropriate URL in case the dataset is remotely accessed (e.g., via the ?SPECS-EUPORIAS THREDDS).

Value

The output of the function consists of a list of variable length, depending on the number of variables contained in the dataset, following this structure:

- `Description`: Description of the variable
- `DataType`: Character string indicating data type (i.e. float ...)
- `Units`: Character string indicating the units of the variable
- `TimeStep`: A `difftime` class object representing the time interval between consecutive values in the time dimension axis

  `Dimensions`: A list of length *n*, containing the following information for each of the *n* dimensions:
  - `Type`: Character vector indicating the type of dimension (e.g. Time, Lon, Pressure ...)
  - `Units`: Character vector indicating the units of the dimension axis
  - `Values`: A vector containing all the dimension values. This might be a vector of `POSIXlt` class in case of a dimension of type *time*, or numeric in other cases.

Details

A common need prior to data analysis is to get an overview of all data available and their structure (variables, dimensions, units, geographical extent, time span ...). Note that the function provides an overview of the raw data as they are stored in the original data files. The units may change after loading the function if conversions are applied via dictionary.

Examples

An example of this function is provided in the Examples section?

loadObservations

Description

Loads observational station data from standard station datasets stored in *.csv* files.

Usage

```
loadObservations(source.dir, var, standard.vars=TRUE, stationID, startDate=NULL, endDate=NULL, season=NULL)
```

Arguments

- `source.dir`: Character string indicating the full path to the directory where the data are stored (see Details).
- `var`: Character string indicating the variable to load.
- `standard.vars`: Logical. Default to `TRUE`. Note that if `TRUE`, a dictionary must be available (see Details)
- `stationID`: Character vector indicating the identification code of the stations to load.
- `startDate`: Optional character string in the form "Year-month-day" (e.g. `"1950-01-01"`) indicating the starting day of the time series to retrieve. Default to first record available.

- `endDate`: Optional character string in the form "Year-month-day" (e.g. `"2000-12-31"`) indicating the last day of the time series to retrieve. Default to last record available.
- `season`: Optional. A vector of integers specifying the desired season (in months, January=1 ...). Options include one to several months. If `NULL` (the default), the function will return all records within the interval defined by `startDate` and `endDate`. For instance, `season = c(12,1,2)` will retrieve the data series for the standard boreal winter (DJF), period = 6:8 for summer (JJA) and so on. See details.

Details

This function works with standard *.csv* observational datasets. It allows loading data from one or several stations at a time. The dictionary is the table that translates the variable as stored in the dataset to the standard variables defined in the vocabulary. More details ?here

In the case of boreal winter selection (`season=c(12,1,2)`) the function will tie strictly to the time interval defined by the `startDate`and `endDate` arguments, and therefore will not retrieve data from the previous December, nor from the next January and February before/after the start/end years defined (note that this has a different behavior than `loadSeasonalForecast` and `loadGCM`)

Value

A list with the containing the following elements:

- `StationID`: character vector with the identification codes of the stations loaded.
- `LatLonCoords`: 2D matrix with the Lat-Lon coordinates of the stations loaded (the order of the rows corresponds to the order of the `StationID` field).
- `Altitude`: numeric vector with the altitude of the stations.
- `Dates`: A `POSIXlt` class vector with the dates of the time series.
- `Data`: 2D matrix with the time series for each of the stations arranged by columns.

Note

An option to read station data from NetCDF files is currently under development and it is expected to be available by the next release of `meteoR`.

Examples

An example of this function is provided in the Examples section?

loadGCM

Description

Loads selected dimensional slices of a NcML dataset. The function is intended to deal with gridded data (interpolated surfaces, reanalysis, RCMs/GCMs ...)

Usage

```
loadGCM(dataset, var, standard.vars=TRUE, dictionary=NULL, lonLim=NULL, latLim=NULL, level=NULL, season=NULL, years=NULL)
```

Arguments

- `dataset`: Character string with the full path to the target ncml file
- `var`: Character string indicating the variable to load.
- `standard.vars`: Logical. Default to `TRUE`. In this case, a dictionary must be available.
- `dictionary`: Character string with the full path to the dictionary. This is used only when `standard.vars = TRUE`. In this case, by default (`dictionary = NULL`), the dictionary is automatically searched in the same directory as the `dataset`, as a file with the same name than the dataset and extension *.dic*.
- `lonLim`: Vector of length = 2, with minimum and maximum longitude coordinates, in decimal degrees, of the bounding box selected. Alternatively, point selection is performed indicating the longitude as a single numeric value. If `NULL` (the default), the whole longitudinal range of the dataset is selected. See details.
- `latLim`: Vector of length = 2, with minimum and maximum latitude coordinates, in decimal degrees, of the bounding box selected. Alternatively, point selection is performed indicating the latitude as a single numeric value. If `NULL` (the default), the whole latitudinal range of the dataset is selected. See details.

- `level`: Vertical level, if defined for the variable. See details
- `season`: Optional. A vector of integers specifying the desired season (in months, January=1 ...). Options include one to several months. If `NULL` (the default), the function will return all records for the years selected.
- `years`: an integer vector specifying the years to select. Default to all years available in the dataset.

Details

The function can select the whole spatial domain covered by the dataset, spatial windows defined by the minimum and maximum corner coordinates, and single grid-cell values. In the last two cases, the function operates by finding the closest grid-points to the coordinates introduced.

For variables with different vertical levels, only defined level values will be allowed, otherwise getting an error. The function does not look for the closest level to the value introduced, in order to avoid confusions. The function `dataInventory` is useful for finding the valid level values defined for a particular variable.

The behavior of the function for year-crossing seasons (e.g. DJF) is similar to `loadSeasonalForecast`.

Value

A list with the following components:

- `VarName`: name of the variable returned.
- `Level`: Vertical level requested. `NULL` if the variable has no vertical levels defined.
  `Dates`: A list with two POSIXlt time elements of length *i*, corresponding to the rows matrix in `Data`. The list contains two elements:
  - `Start`: Starting times of the verification period of the variable
  - `End`: End time of the verification period of the variable
- `LatLonCoords`: A 2-D matrix of *j* rows (where *j* = number of grid points selected) and two columns corresponding to the latitude and longitude coordinates respectively.
- `Data`: a 2-D matrix of *i* rows x *j* columns, of *i* times and *j* grid-points

Examples

An example of this function is provided in the Examples section?

loadSeasonalForecast

Description

Loads seasonal hindcast/forecast data from the SPECS-EUPORIAS THREDDS Data Server.

Usage

```
loadSeasonalForecast(dataset, var, standard.vars = TRUE, dictionary = NULL, members, lonLim, latLim, season, years, leadMo
```

Arguments

- `dataset`: A character string indicating the full URL path to the OPeNDAP dataset. Currently, the accepted values correspond to the System4 ?available datasets at the SPECS-EUPORIAS THREDDS Data Server.
- `var`: Variable code. This is the name of the variable either as coded in the dataset (as provided by the data inventory) or according to the `identifier` code in the dictionary if `standard.vars = FALSE` or `TRUE` respectively.
- `standard.vars`: Logical. Default to `TRUE`. In this case, a dictionary must be available.
- `dictionary`: Character string with the full path to the dictionary. This is used only when `standard.vars = TRUE`. In this case, by default (`dictionary = NULL`), the dictionary is automatically searched in the same directory as the dataset, as a file with the same name than the dataset and extension *.dic*.
- `members`: Optional. A vector of integers indicating the members to be loaded. Default to `NULL`, which loads all members available. For instance, `members=1:5` will retrieve the first five members.
- `lonLim`: Vector of length = 2, with minimum and maximum longitude coordinates, in decimal degrees, of the bounding box selected. For single-point queries, a numeric value with the longitude coordinate. See details.
- `latLim`: Vector of length = 2, with minimum and maximum latitude coordinates, in decimal degrees, of the bounding box selected. For single-point queries, a numeric value with the latitude coordinate. See details.

- `season`: A vector of integers specifying the desired season (in months, January=1 ...). Options include one to several months. If `NULL` (the default), the function will return all possible months given the `leadMonth`. For instance, `period = c(12,1,2)` will retrieve the forecast for the standard boreal winter (DJF), `period = 6:8` for summer (JJA) and so on. See details.
- `years`: Optional vector of years to select. Default to all available years. Note that in the case of a year-crossing season for a particular year period (e.g. winter DJF, `season = c(12,1,2)` and `years = 1981:2000`), by convention the first season returned will be DJF 1980/81, if available (otherwise a warning message is given). See details.
- `leadMonth`: Lead month forecast time corresponding to the first month of the specified season. Note that `leadMonth = 1` for `season = 1` (January) corresponds to the December initialization forecasts. The effect of the lead time in the forecast for a particular season can be analyzed by just changing this parameter.

Details

Regarding the spatial component of the returned data, the function can select the whole spatial domain covered by the dataset (in this case `lonLim = NULL` and `latLim = NULL`), spatial windows defined by the minimum and maximum corner coordinates (for instance `lonLim = c(-10,10)` and `latLim = c(35,45)` indicates a rectangular window), and single grid-cell values (for instance `lonLim = -3.21` and `latLim = 41.087` for retrieving the data in the closest grid point to the point coordinate -3.21E , 41.087N). In the last two cases, the function operates by finding the nearest (euclidean distance) grid-points to the coordinates introduced.

The function has been implemented to access seasonal slices (as determined by the `season` argument. Seasons can be defined in several ways: A single month (e.g. `season = 1` for January), a standard season (e.g. `season=c(1,2,3)` for JFM, or `season=c(12,1,2)` for DJF), or any period of consecutive months (e.g. `season=c(1:6)`, for the first half of the year). Seasons are returned for a given year period (defined by the `years` argument, e.g. `years = 1981:2000`) with a homogeneous forecast lead time (as given by the `leadMonth` argument; e.g. `leadMonth = 1` for one-month lead time) with respect to the first month of the selected season. For example, `season=c(1,2,3)` for `years = 1995:2000` and `leadMonth = 1` will return the following series: JFM 1995 from the December 1994 runtime forecast, ..., JFM 2000 from the December 1999 runtime forecast. Note that it is also possible to work with year-crossing seasons, such as DJF. In this case, `season=c(12,1,2)` for `years = 1995:2000` and `leadMonth = 1` will return the following series: DJF 1994/1995 (from the November 1994 runtime forecast), ..., DJF 1999/2000 (from the November 1999 runtime forecast).

Value

The output returned by the function consists of a list with the following elements providing the necessary information for data representation and analysis:

- `VarName`: A character string indicating which is the variable returned. Same as value provided for argument `var`.
- `isStandard`: Logical value indicating whether the variable returned is standard or not. Same as value provided for argument `standard.vars`
- `MemberData`: This is a list of length *n*, where *n* = number of members of the ensemble selected by the `member` argument. Each element of the dataset is a 2-D matrix of *i* rows x *j* columns, of *i* forecast times and *j* grid-points
- `LatLonCoords`: A 2-D matrix of *j* rows (where *j* = number of grid points selected) and two columns corresponding to the latitude and longitude coordinates respectively.
- `RunDates`: A `POSIXlt` time object corresponding to the initialization times selected. There is an initialization time associated to each forecast time.

  `ForecastDates`: A list with two `POSIXlt` time elements of length *i*, corresponding to the rows of each matrix in `MemberData`. The list contain tow elements:
  - `Start`: Starting times of the verification period of the variable
  - `End`: End time of the verification period of the variable

Examples

An example of this function is provided in the Examples section?

makeNcmlDataset

Description

Generates a NcML file from a collection of netCDF files.

Usage

```
makeNcmlDataset(source.dir, ncml.file)
```

Arguments

- `source.dir`: character string indicating a valid path of the directory containing the files
- `ncml.file`: character string indicating the NcML file name (and path, default to working directory), including the extension *.ncml*.

The output is a NcML file named as `file.name` which will be stored in the `output.dir`.

Details

- All files of the same dataset should be put together in the same directory, indicated by the `source.dir` argument.
- Currently the function works only with netCDF (*.nc*) file collections.
- A number of useful recommendations regarding dataset naming are provided ?here

Value

Creates a NcML file at the specified location

Notes

A NcML file is a ?XML representation of netCDF metadata. This is approximately the same information one gets when dumping the header of a netCDF file (e.g. by typing on the terminal the command `ncdump -h`). By means of NcML it is possible to create virtual datasets by modifying and aggregating other datasets, thus providing maximum flexibility and ease of access to data stored in collections of files containing data from different variables/time slices. The function `makeNcmlDataset` is intended to deal with reanalysis, forecasts and other climate data products, often consisting of collections of netCDF files corresponding to different variables and partitioned by years/decades or other time slices. It operates by applying to types of ?aggregation operations:

1. `Union`: Performs the union of all the dimensions, attributes, and variables in multiple NetCDF files
2. `JoinExisting`: Variables of the same name (in different files) are connected along their existing, outer dimension, called the aggregation dimension. In this case the aggregation dimension is *time*.

Examples

An example of this function is provided in the Examples section?