Package ‘meteoForecast’

February 20, 2015

**Type**  Package
**Title**  Numerical Weather Predictions
**Version**  0.43
**Date**  2014-10-17

**Description**
Access to several Numerical Weather Prediction services both in raster format and as a time series for a location. Currently it works with GFS, MeteoGalicia, OpenMeteo, NAM, and RAP.

**URL**  httpZOOgithubNcomOoscarperpinanOmeteoforecast
**BugReports**  https://github.com/oscarperpinan/meteoForecast/issues
**License**  GPL-3

**Depends**  raster, sp, zoo
**Imports**  ncdf, rjson
**Suggests**  ncdf4, rgdal, lattice, rasterVis

**Author**  Oscar Perpinan Lamigueiro [cre, aut], Marcelo Pinho Almeida [ctb]
**Maintainer**  Oscar Perpinan Lamigueiro <oscar.perpinan@gmail.com>
**NeedsCompilation**  no

**Repository**  CRAN
**Date/Publication**  2014-10-19 22:28:02

**R topics documented:**

- meteoForecast-package .................................................. 2
- Forecast variables ...................................................... 2
- getPoint ........................................................................ 5
- getRaster ....................................................................... 7
- options ........................................................................ 11

**Index**  12
Description

The Weather Research and Forecasting (WRF) Model is a numerical weather prediction (NWP) system. NWP refers to the simulation and prediction of the atmosphere with a computer model, and WRF is a set of software for this.

meteoForecast downloads data from the MeteoGalicia and OpenMeteo NWP-WRF services using the NetCDF Subset Service.

Details

getRaster, getRasterDay, and getRasterDays get data inside a bounding box and provide a multilayer raster data using the RasterBrick class defined in the package raster.

getPoint, getPointDays, and getPointRuns get data for a certain location and produce a time series using the zoo class.

Author(s)

Oscar Perpiñán, with contributions from Marcelo Almeida

References

http://www.wrf-model.org/index.php
http://www.meteogalicia.es/web/modelos/threddsIndex.action
https://openmeteoforecast.org/wiki/Main_Page

See Also

raster zoo

Forecast variables

Forecast Variables available in each model.

Description

Variables available in each model. The grepVar function is a wrapper around grep to search for variables.
Forecast variables

Usage

grepvar(x, service, complete = FALSE)
data(varsMG)
data(varsOM)
data(varsGFS)
data(varsNAM)
data(varsRAP)

Arguments

x                character string to be matched in the description column of the corresponding vars* file.
service          Character, to choose from 'meteogalicia', 'openmeteo', 'gfs', 'nam', and 'rap'
complete         Logical, if FALSE (default) only the name column is returned. If TRUE the three columns are provided.

Format

Each data.frame contains three columns, name, label, and description, with the information about the variables available in each service. Use the elements of the name column to choose a variable with the argument var of getRaster and getPoint.

Source

varsMG: http://www.meteogalicia.es/web/modelos/threddsIndex.action
varsOM: https://openmeteoforecast.org/wiki/Main_Page
varsGFS: http://nomads.ncdc.noaa.gov/thredds/catalog/gfs-004/catalog.html
varsNAM: http://nomads.ncdc.noaa.gov/thredds/catalog/nam218/catalog.html
varsRAP: http://nomads.ncdc.noaa.gov/thredds/catalog/rap130/catalog.html

Examples

grepVar('cloud', service = 'gfs', complete = TRUE)
data(varsMG)
head(varsMG)
data(varsOM)
head(varsOM)
data(varsGFS)
head(varsGFS)
data(varsNAM)
head(varsNAM)
data(varsRAP)
head(varsRAP)

## Not run:
## This information has been extracted using this code
library(XML)

## MeteoGalicia
wcs <- xmlParse(paste0(
  'http://mandeo.meteogalicia.es/thredds/wcs/modelos/WRF_HIST/d02/',
  '2014/07/wrf_arw_det_history_d02_20140721_0000.nc4',
  '?service=WCS&version=1.0.0&request=GetCapabilities'))
doc <- xmlRoot(wcs)
content <- xmlChildren(doc)
meta <- content[['ContentMetadata']]
xmlSize(meta)

varsMG <- xmlToDataFrame(meta)[, c('description', 'name', 'label')]

## GFS
wcs <- xmlParse(paste0(
  'http://nomads.ncep.noaa.gov/thredds/wcs/gfs-004/',
  '201407/20140715/gfs_4_20140715_1200_165.grb2',
  '?service=WCS&version=1.0.0&request=GetCapabilities'))
doc <- xmlRoot(wcs)
content <- xmlChildren(doc)
meta <- content[['ContentMetadata']]
xmlSize(meta)

varsGFS <- xmlToDataFrame(meta)[, c('description', 'name', 'label')]

## NAM
wcs <- xmlParse(paste0(
  'http://nomads.ncep.noaa.gov/thredds/wcs/nam218/',
  '201408/20140802/nam_218_20140802_0000_001.grb',
  '?service=WCS&version=1.0.0&request=GetCapabilities'))
doc <- xmlRoot(wcs)
content <- xmlChildren(doc)
meta <- content[['ContentMetadata']]
xmlSize(meta)

varsNAM <- xmlToDataFrame(meta)[, c('description', 'name', 'label')]

## RAP
wcs <- xmlParse(paste0(
  'http://nomads.ncep.noaa.gov/thredds/wcs/rap130/',
  '201408/20140802/rap_130_20140802_0000_001.grb2',
  '?service=WCS&version=1.0.0&request=GetCapabilities'))
doc <- xmlRoot(wcs)
content <- xmlChildren(doc)
meta <- content[['ContentMetadata']]
xmlSize(meta)

varsRAP <- xmlToDataFrame(meta)[, c('description', 'name', 'label')]
**Description**

The `getPoint` functions get outputs of the NWP models run by MeteoGalicia, NCDC (GFS), OpenMeteo for a single location.

**Usage**

```r
getPoint(point, vars = "swflx", day = Sys.Date(), run = "00",
          resolution = NULL, service = mfservice())

getPointDays(point, vars = "swflx", start = Sys.Date(), end,
              resolution = NULL, service = mfservice())

getPointRuns(point, var = "swflx",
             start = Sys.Date() - 1, end = Sys.Date(),
             resolution = NULL, service = mfservice())
```

**Arguments**

- **point**: Coordinates of the location. It can be a `SpatialPoints` or a numeric of length 2.
- **var, vars**: Character. The name of the variables to retrieve. Inspect `varsMG`, `varsOM`, and `varsGFS` to know what variables are available in each service. `getPointRuns` only works with one variable.
- **day**: Date or character
- **run**: Character. The meteogalicia service executes the model at OUTC and 12UTC. Therefore `run` can be '00' or '12'. With OpenMeteo and GFS `run` can be '00', '06', '12', and '18'.
- **start**: Date or character. First day of the time period to retrieve.
- **end**: Date or character. Last day of the time period to retrieve.
- **resolution**: Numeric. Resolution in kilometers of the raster. Valid choices are 4, 12, and 36. It is only used with `service = 'meteogalicia'`.
- **service**: Character, which service to use, 'meteogalicia', 'gfs', or 'openmeteo'. Please note that the support for the 'gfs' and 'openmeteo' services is still experimental and limited.
getPoint

Details

These functions download data from the MeteoGalicia, NCDC, OpenMeteo servers using the NetCDF Subset Service. The result is returned as a zoo time series object, with one or more csv files stored in the temporary folder (as defined by tempdir()).

Value

getPoint and getPointDays produce a zoo time series with a column for each variable included in vars.

The time series returned by getPoint starts at 01UTC of day if run = '00' or 13UTC if run = '12'. It spans over 4 days (96 hours) if run = '00' or 84 hours if run = '12'.

The time series returned by getPointDays starts at 01UTC of start and finishes at 00UTC of end + 1. Each day comprised in the time period is constructed with the forecast outputs corresponding to the 00UTC run of that day. Therefore, only the first 24 values obtained with getPoint are used for each day.

The time series returned by getPointRuns starts at 01UTC of start and finishes at 00UTC of end + 1. It has 4 columns, named "D3_00", "D2_00", "D1_00" and "D0_00". The column "D3_00" corresponds to the forecast results produced 3 days before the time stamp of each row, and so on.

Author(s)

Oscar Perpiñán Lamigueiro with contributions from Marcelo Almeida

References

http://www.meteogalicia.es/web/modelos/threddsIndex.action
https://openmeteoforecast.org/wiki/Main_Page
http://nomads.ncdc.noaa.gov/thredds/catalog/gfs-004/catalog.html

See Also

getRaster

Examples

## Not run:
## temperature (Kelvin) forecast from meteogalicia
tempK <- getPoint(c(0, 40), vars = 'temp')
## Cell does not coincide exactly with request
attr(tempK, 'lat')
attr(tempK, 'lon')
## Units conversion
tempC <- tempK - 273

library(lattice)
## Beware: the x-axis labels display time using your local timezone.
Sys.timezone()
getRaster

```r
## Use Sys.setenv(TZ = 'UTC') to produce graphics with the timezone
## of the objects provided by meteoForecast.
xyplot(tempC)

## Multiple variables
vars <- getPoint(c(0, 40), vars = c('swflx', 'temp'))
xyplot(vars)

## Time sequence
radDays <- getPointDays(c(0, 40), start = '2013-01-01',
                        end = '2013-01-15')

xyplot(radDays)

## Variability between runs
radRuns <- getPointRuns(c(0, 40), start = '2013-01-01',
                        end = '2013-01-15')

xyplot(radRuns, superpose = TRUE)

## Variability around the average
radAv <- rowMeans(radRuns)
radVar <- sweep(radRuns, 1, radAv)

xyplot(radVar, superpose = TRUE)

## End(Not run)
```

---

**getRaster**

*NWP forecasts for a region*

**Description**

The getRaster* functions get outputs of the NWP models for a region.

**Usage**

getRaster(var = "swflx", day = Sys.Date(), run = "00",
          frames = 'complete', box, resolution = NULL,
          names, remote = TRUE, service = mfService(),
          dataDir = ".", use00H = FALSE, ...)

getRasterDays(var = "swflx", start = Sys.Date(), end,
              remote = TRUE, dataDir = ".", ...)

getRasterDay(var = "swflx", day = Sys.Date(),
             remote = TRUE, dataDir = ".", ...)

checkDays(start, end, vars, remote = FALSE,
          service = mfService(), dataDir = ".")
**Arguments**

- **var, vars**: Character. The name of the variable (or variables in checkDays) to retrieve. Inspect the help page of `grepVar` to know what variables are available in each service.

- **day**: Date or character. In `getRaster` it defines the day when the forecast was produced. In `getRasterDay` it defines the day to be forecast.

- **run**: Character. For example, the meteogalicia service executes the model at OOUTC and 12UTC. Therefore run can be '00' or '12'.

- **start, end**: Date or character. First day of the time period to retrieve. Last day of the time period to retrieve.

- **frames**: Numeric. It defines the number of hourly forecasts (frames) to retrieve. If frames = 'complete', the full set of frames is downloaded. For example, the meteogalicia service produces 96 hourly forecasts (frames) with run='00' and 84 frames with run='12'.

- **box**: The bounding box, defined using longitude and latitude values. A Extent or an object that can be coerced to that class with extent: a 2x2 matrix (first row: xmin, xmax; second row: ymin, ymax), vector (length=4; order= xmin, xmax, ymin, ymax) or list (with at least two elements, with names 'x' and 'y').

- **resolution**: Numeric. Resolution in kilometers of the raster. Valid choices are 4, 12, and 36. It is only used with service = 'meteogalicia'.

- **names**: Character. Names of the layers of the resulting RasterBrick. If missing, a predefined vector is assigned the combination of day and hour.

- **remote**: Logical. If TRUE (default) data is downloaded from the remote service. If FALSE the RasterBrick is produced with the files available in the local folder.

- **service**: Character, which service to use, 'meteogalicia', 'gfs', 'openmeteo', 'nam' or 'rap'. Note that the support for the 'openmeteo' service is still experimental and limited.

- **use00H**: Logical. Only used when service is 'gfs', 'nam', or 'rap'. If FALSE (default), the first frame of each run or 00H "forecast" is not considered. This first frame is only produced for some variables. Therefore, with use00H = TRUE fewer frames that the number defined with frames could be obtained for some variables.)

- **dataDir**: Character, path of the folder where files are stored (if remote = 'FALSE')

- ... Additional arguments. Not used in `getRaster`.

**Details**

`getRaster` downloads data from the MeteoGalicia, NCDC (GFS, RAP, and NAM), or OpenMeteo servers using the NetCDF Subset Service. The result is returned as a RasterBrick object, with one or more NetCDF files stored in the temporary folder (as defined by `tempdir()`). Each frame or layer of the RasterBrick corresponds to a certain hour of the forecast.

`getRasterDay` uses `getRaster` to download the results corresponding to a certain day. If the day is in the future, the most recent forecast is downloaded with `getRaster`, and the corresponding frames are extracted. If the day is in the past, `getRaster` is used to download the corresponding frames of the forecast produced that day.
getRasterDays uses getRaster to download the results cast each day comprised between start and end using the 00UTC run. Then it subsets the first 24 frames of each result, and binds them together to produce a RasterBrick. Therefore, each frame of this RasterBrick is a forecast for an hour of the day when the forecast was cast.

checkDays explores a local folder looking for NetCDF files corresponding to a time sequence and a set of variables. It returns a Date vector comprising the days with files available for the requested variables. If remote = TRUE it only checks that start is after 2008-01-01 (first date of the archived forecasts of MeteoGalicia.)

Value

The getRaster* functions return a RasterBrick with a layer for each hour of the NWP forecast. The time zone of the time index of this RasterBrick, stored in its z slot (accessible with getZ) is UTC.

MeteoGalicia, OpenMeteo, NAM, and RAP use the Lambert Conic Conformal projection. GFS files use longitude-latitude coordinates.

Author(s)

Oscar Perpiñán with contributions from Marcelo Almeida.

References

http://www.meteogalicia.es/web/modelos/threddsIndex.action
https://openmeteoforecast.org/wiki/Main_Page
https://openmeteoforecast.org/wiki/OPeNDAP_Server
http://nomads.ncdc.noaa.gov/thredds/catalog/gfs-004/catalog.html
http://nomads.ncdc.noaa.gov/thredds/catalog/rap130/catalog.html
http://nomads.ncdc.noaa.gov/thredds/catalog/nam218/catalog.html

Examples

## Not run:
library(rasterVis)

## Retrieve raster data
wrf <- getRaster('temp', '2014-01-25', '00', remote=TRUE)

levelplot(wrf, layers = 10:19)

hovmoller(wrf)

## Using box and frames specification
mfExtent('gfs')
cloudGFS <- getRaster('Total_cloud_cover',
          day = '2014-10-01',
          box = c(-10, 10, -10, 10),
          service = 'gfs')
```r
mfExtent('nam')
cloudNAM <- getRaster('Total_cloud_cover',
day = '2014-10-01',
box = c(-100, -80, 30, 50),
frames = 10,
service = 'nam')

mfExtent('rap')
cloudRAP <- getRaster('Total_cloud_cover',
day = '2014-10-01',
box = c(-100, -80, 30, 50),
frames = 10,
service = 'rap')

## Day sequence of cloud cover
wrfDays <- getRasterDays('cft',
start = '2014-01-01',
end = '2014-01-05',
box = c(-2, 35, 2, 40))

levelplot(wrfDays, layers = 10:19)

## animation
levelplot(wrfDays, layout = c(1, 1), par.settings = BTCTheme)

## Höv moller graphic
hovmoller(wrfDays, par.settings = BTCTheme, contour = TRUE, cuts = 10)

NAMDays <- getRasterDays('Total_cloud_cover',
start = '2014-01-01',
end = '2014-01-03',
box = c(-100, -80, 30, 50),
service = 'nam')

## Extract data at some locations
st <- data.frame(name=c('Almeria','Granada','Huelva','Malaga','Caceres'),
elev=c(42, 702, 38, 29, 448))

coordinates(st) <- cbind(c(-2.46, -3.60, -6.94, -4.42, -6.37),
c(36.84, 37.18, 37.26, 36.63, 39.47))
proj4string(st) <- '+proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0'

## Extract values for some locations
vals <- extract(wrf, st)
vals <- zoo(t(vals), getZ(wrf))
names(vals) <- st$name

xyplot(vals)
```

Options

Description
Functions to get or set options, and to access internal parameters of the package.

Usage

getMFOption(name = NULL)
setMFOption(name, value)
mService(service = NULL)
mfExtent(service, resolution = 12)
mProj4(service)

Arguments

name Character, name of the option to get or set.
value Character, value of the option to be changed.
service Character, name of the service.
resolution Numeric, value of the resolution (in kilometers). Only useful if service = 'meteogalicia'

Details
Use getMFOption to list the options of the package. Only one option, service, is available with this version. With setMFOption the option defined with name can be modified.
mService, a wrapper around getMFOption and setMFOption, displays the default service if used without arguments. It modifies the default service to the value of its argument.
mfExtent and mProj4 provides the extent and the proj4 string of the corresponding service.

Author(s)
Oscar Perpiñán Lamigueiro

Examples

mService()

mfExtent('meteogalicia', 36)
mfExtent('nam')
mProj4('rap')
## Index

<table>
<thead>
<tr>
<th>Topic</th>
<th>Packages/Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>datasets</strong></td>
<td>Forecast variables, 2</td>
</tr>
<tr>
<td><strong>package</strong></td>
<td>meteoForecast-package, 2</td>
</tr>
<tr>
<td><strong>raster</strong></td>
<td>getRaster, 7</td>
</tr>
<tr>
<td><strong>spatial</strong></td>
<td>getPoint, 5, getRaster, 7</td>
</tr>
<tr>
<td><strong>time series</strong></td>
<td>getPoint, 5</td>
</tr>
<tr>
<td>checkDays</td>
<td>(getRaster), 7</td>
</tr>
<tr>
<td>Extent</td>
<td>8</td>
</tr>
<tr>
<td>extent</td>
<td>8</td>
</tr>
<tr>
<td>Forecast variables</td>
<td>2</td>
</tr>
<tr>
<td>getMFOption</td>
<td>(options), 11</td>
</tr>
<tr>
<td>getPoint</td>
<td>3, 5</td>
</tr>
<tr>
<td>getPointDays</td>
<td>(getPoint), 5</td>
</tr>
<tr>
<td>getPointRuns</td>
<td>(getPoint), 5</td>
</tr>
<tr>
<td>getRaster</td>
<td>3, 6, 7</td>
</tr>
<tr>
<td>getRasterDay</td>
<td>(getRaster), 7</td>
</tr>
<tr>
<td>getRasterDays</td>
<td>(getRaster), 7</td>
</tr>
<tr>
<td>getZ</td>
<td>9</td>
</tr>
<tr>
<td>grepVar</td>
<td>8</td>
</tr>
<tr>
<td>grepVar</td>
<td>(Forecast variables), 2</td>
</tr>
<tr>
<td>meteoForecast</td>
<td>(meteoForecast-package), 2</td>
</tr>
<tr>
<td>meteoForecast-package</td>
<td>2</td>
</tr>
<tr>
<td>mfExtent</td>
<td>(options), 11</td>
</tr>
<tr>
<td>mfProj4</td>
<td>(options), 11</td>
</tr>
<tr>
<td>mfService</td>
<td>(options), 11</td>
</tr>
<tr>
<td>options</td>
<td>11</td>
</tr>
<tr>
<td>raster</td>
<td>2</td>
</tr>
<tr>
<td>setMFOption</td>
<td>(options), 11</td>
</tr>
</tbody>
</table>

varsGFS, 5
varsGFS | (Forecast variables), 2 |
varsMG, 5
varsMG | (Forecast variables), 2 |
varsNAM | (Forecast variables), 2 |
varsOM, 5
varsOM | (Forecast variables), 2 |
varsRAP | (Forecast variables), 2 |
zoo, 2