Technical Notes
Santander Meteorology Group (CSIC-UC)
http://www.meteo.unican.es
SMG:01.2014

Introduction to the ECOMS User Data Gateway `R` interface

J. Bedia[1], M.E. Magariño[2], S. Herrera[2], R. Manzanas[1], J. Fernández[2], A.S. Cofiño[2] & J.M. Gutiérrez[1]
[1] *Instituto de Física de Cantabria, CSIC-Universidad de Cantabria, Spain.*
[2] *Dpto. de Matemática Aplicada y C.C. Universidad de Cantabria, Spain*
correspondence: bediaj@unican.es

version:v1.0–*31 Jan 2014*

Abstract

The ECOMS Data User Gateway (ECOMS UDG), is based on a password-protected THREDDS data server providing metadata and data access to a set of georeferenced atmospheric variables using OPeNDAP and other remote data access protocols. The `ecomsUDG.Raccess` library is intended as a user-friendly interface for accessing the datasets stored at the ECOMS UDG, relying on the powerful capabilities of the Unidata's netCDF Java library.

The ECOMS UDG wiki: https://www.meteo.unican.es/trac/wiki/EcomsUdg

1 ECOMS UDG Registration

As different usage policies and conditions apply to the various datasets stored at the ECOMS UDG, a fine-grained user authorization scheme has been implemented to adapt to all of them. The applicants for an ECOMS UDG authorization must accept the usage terms and conditions for the different dataset access roles.

The steps to follow in order to request a username and a password are indicated in the ECOMS UDG wiki page.

1.1 User authentication

Once a valid user name (e.g. "myUser") and password are issued (e.g. "myPassword"), HTTP authentication is directly achieved within a R session using the function `loginECOMS_UDG`:

```
> library(ecomsUDG.Raccess)
> loginECOMS_UDG(username = "myUser", password = "myPassword")
```

In case the connection is done via a proxy server, the name of the server and the proxy port must be provided filling the corresponding arguments. Type `?loginECOMS_UDG` for details.

2 Accesing Data

2.1 Data homogeneization

The diverse characteristics of the different climate products, models and variables, and the idiosyncratic naming and storage conventions often applied by the various modelling centres, requires previous dataset homogeneization. The `ecomsUDG.Raccess` package pursues this aim by defining a common *vocabulary* to all datasets. The variables of each particular dataset are translated —and transformed when necessary— to the common vocabulary by means of a *dictionary*, which contains the necessary information to unequivocally define the time aggregation of the data and the conversion operations needed to get the standard units.

The vocabulary is included as a dataset in the `ecomsUDG.Raccess` package. So far, only the currently available variables are included, although the vocabulary is subject to continuous update along with the ECOMS UDG datasets. Thus, the vocabulary defines the "standard variables".

```
> data(vocabulary)
> print(vocabulary)
  identifier              standard_name           units
1        tas        2-meter temperature degrees Celsius
2     tasmax    maximum 2-m temperature degrees Celsius
3     tasmin    minimum 2-m temperature degrees Celsius
4         tp Total precipitation amount              mm
5        psl  air pressure at sea level              Pa
```

Therefore, the ECOMS UDG user do not need to worry about the time aggregation and units of the variables when using the default 'identifier' indicated in the vocabulary as input for the argument `var` in the `loadSeasonalForecast` function. More advanced users interested in obtaining the original model variables can retrieve them easily, as explained in the ECOMS UDG wiki.

2.2 Data retrieval

In the next lines a few examples using the `loadSeasonalForecast` function are presented. These simple examples are intended to give an overview of the data output size and the time needed for download, although the time largely depends on the characteristics of the internet connection. Note that for brevity, in the examples below we have ommitted the screen messages arising from the queries to the server.

More ellaborated worked examples are presented in the ECOMS UDG wiki

EXAMPLE 1: Point Scale Single point locations can be acessed by indicating their geographical coordinates in the `lonLim` and `latLim` arguments. Due to the limited spatial domain of this type of queries, this option allows retrieving long time series for all ensemble members without worrying for the memory size of the returned object. In this example we load the 15 members for the System4 seasonal model and summer (JJA) maximum 2m temperature data for the whole model time span (30 years) at Madrid (Spain, -3.680E, 40.40N):

```
> system.time(
+  ex1 <- loadSeasonalForecast("System4_seasonal_15", var = "tasmax",
+ dictionary = TRUE, lonLim = -3.7, latLim = 40.4, season = 6:8,
+ leadMonth = 1)
+ )
INFO: basic authentication scheme selected
Jan 31, 2014 12:17:57 PM org.apache.commons.httpclient.auth.AuthCha ...
...
Done
```

```
   user  system elapsed
 10.740   0.344 104.192
> # Approximately 2 minutes
> object.size(ex1)
682268 bytes
```

EXAMPLE 2: National/Regional Scale The next example retrieves the 1 month forecast of minimum 2m temperature data for boreal winter (DJF) for the Iberian Peninsula, considering the 10-year period 1991-2000 and all 51 ensemble members:

```
> system.time(
+ ex2 <-  loadSeasonalForecast("System4_seasonal_51", var="tasmin",
+ dictionary=TRUE, members=NULL, lonLim=c(-10,10), latLim=c(35,44),
+ season=c(12,1,2), years=1991:2000, leadMonth=1)
+)
Jan 31, 2014 1:04:04 PM org.apache.commons.httpclient.auth.AuthCha ...
INFO: basic authentication scheme selected
Done
   user  system elapsed
 24.581   1.552 392.566
> print(object.size(ex2), units = "Mb")
114 Mb
```

Data loading took around 7 minutes, and the object returned is more than 100 Mb. Note that we have requested 51 members, which generates a quite large object.

EXAMPLE 3: Continental Scale In the case of larger spatial domains, it is recommended to introduce shorter time periods or avoiding the selection of multiple members. In case larger datasets are needed, the job should be divided in different calls to the function.

This is an example for the forecasted Spring precipitation (MAM) in January (lead month = 3), spanning a 10-year period for the whole European domain, and considering the 5 first members:

```
> system.time(
+ ex3 <- loadSeasonalForecast(dataset="System4_seasonal_15", var="tp",
+ dictionary=TRUE, members=1:5, lonLim=c(-10,50), latLim=c(35,70),
+ season=c(12,1,2), years=2001:2010, leadMonth=3)
+)
Jan 31, 2014 2:39:39 PM org.apache.commons.httpclient.auth.AuthCha ...
INFO: basic authentication scheme selected
Done
   user  system elapsed
  6.277   0.816  64.387
> print(object.size(ex3), units = "Mb")
129.5 Mb
```

The request took 1 minute and the object size is 130 Mb. In the following figure the spatial domain of the request is represented.

```
> data(world_map)
> plot(ex3$LonLatCoords, axes = TRUE, col = "grey", asp =1)
> lines(as(world_map, "SpatialLines"), col = "red")
```

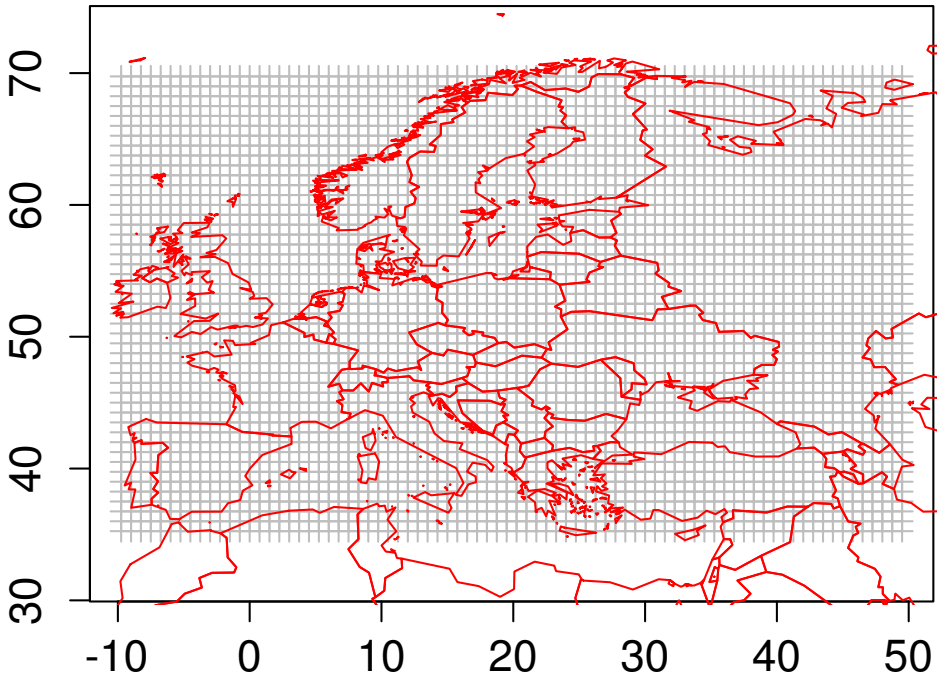

Figure 1: Spatial domain of example 3.

EXAMPLE 4: Global Scale A selection of the whole spatial domain can be also done, by setting the `lonLim` and `latLim` arguments to their default value (`NULL`), although this is not recommended. Due to the large size of the data requested in this case, the user will have to select short time periods and a few or a single members for each call to the function.

```
> system.time(
+ ex4 <- loadSeasonalForecast(dataset="CFSv2_seasonal_16", var="tp",
+ dictionary=TRUE, members=1, season=1, years=2010, leadMonth=3)
+ )
[2014-01-31 16:06:37] Retrieving data from member 1 out of 1...
Done
   user  system elapsed
  3.928   0.780  13.408
> print(object.size(ex4), units = "Mb")
70.2 Mb
> plot(ex4$LatLonCoords, axes = TRUE, cex = .1, col = "grey", pch = ".")
> data(world_map)
> lines(as(world_map, "SpatialLines"), col = "red")
> title(paste(length(ex4$LonLatCoords), "Grid Points"))
```

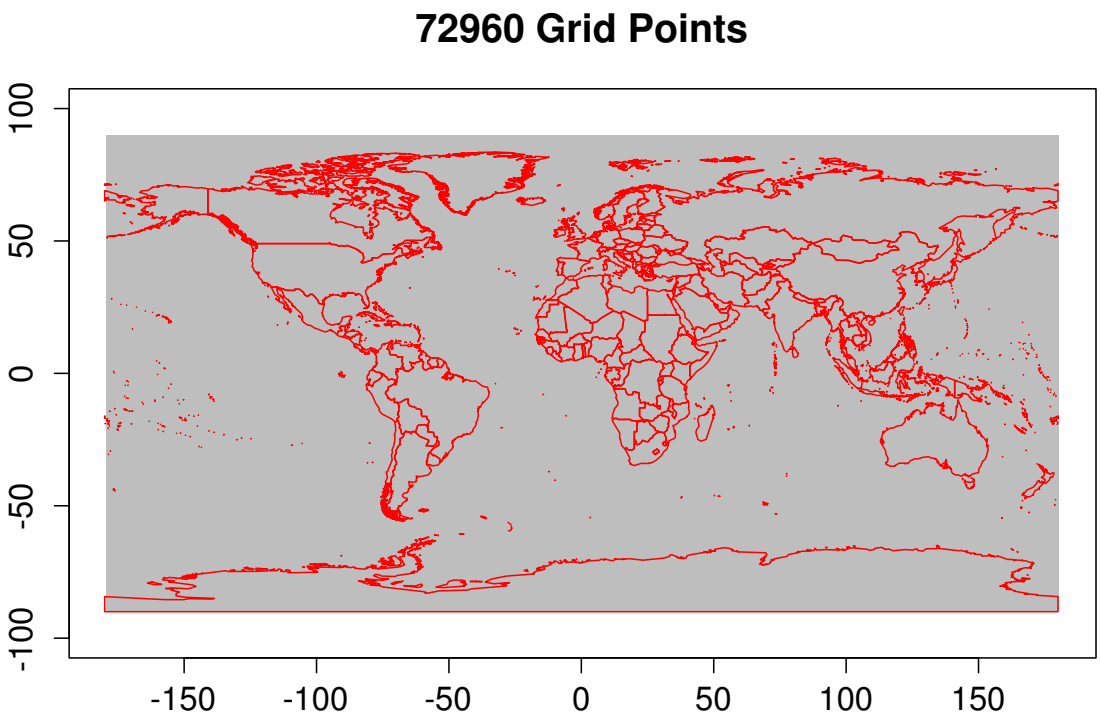

Figure 2: Example 4. Global extent of the CFSv2 model