Single point selections

The following call to `loadECOMS` will load a time series of summer 2001 (JJA, `season = 6:8`, `years = 2001`) surface (2m) daily mean temperature (`var = "tas"`, as defined in the ?vocabulary) for the coordinate -3.7E 40.4N, (`lonLim = -3.7`, `latLim = 40.4`) corresponding to the city of Madrid (Spain), as forecasted the previous March (`leadMonth = 2`) by the CFSv2 hindcast (`dataset = "CFSv2_seasonal_16"`). We will select the first 10 members (`members = 1:10`). Note that the original variable is stored in the CFSv2 database as 6-hourly. Hence, we indicate to the function to compute the daily mean from the 6-hourly data using the argument `time = "DD"`.

First of all, the library is called and we log-in into the ECOMS-UDG:

```
library(ecomsUDG.Raccess)
loginECOMS_UDG("user","password")
```

Now we are ready for accessing the ECOMS-UDG:

```
point.cfs <- loadECOMS(dataset = "CFSv2_seasonal_16", var = "tas", members = 1:10, lonLim = -3.7, latLim = 40.4, season =
```

Note that a number of informative messages will appear on-screen:

```
[2014-09-02 15:28:42] Defining homogeneization parameters for variable "tas"
NOTE: daily mean will be calculated from the 6-h model output
[2014-09-02 15:28:42] Defining geo-location parameters
[2014-09-02 15:28:42] Defining initialization time parameters
[2014-09-02 15:28:46] Retrieving data subset ...
[2014-09-02 15:31:25] Done
```

This is the size of the loaded object:

```
print(object.size(point.cfs)) # 22456 bytes
```

The returned object contains all the necessary information for data representation (geo-location, time ...). In the next lines we plot the loaded time series for each member. The element `Data` contains the data itself. In this case, it is a 2D array with the dimensions `member` (10 members selected) and `time` (92 days for June, July and August), as indicated by the `dimensions` attribute.

```
str(point.cfs$Data)
```

which returns information on the Data array structure:

```
 num [1:10, 1:92] 16.3 17.1 12.9 10.4 15.3 ...
- attr(*, "dimensions")= chr [1:2] "member" "time"
```

Note that, by convention, the dimensions of the Data array will be always ordered in the canonical form *member* > *time* > *lat* > *lon*. Several vertical levels are never loaded at the same time, so the dimension *level* will never appear. This will be indicated in the Variables element of the returned output if existing. In this case it is NULL, because the variable loaded is a surface variable:

```
str(point.cfs$Variable)
```

returns the structure of the Variable element of the output:

```
List of 3
$ varName   : chr "tas"
$ isStandard: logi TRUE
$ level     : NULL
```

In the following example we plot the time series with the multi-member mean and its dispersion (interquartilic range 25-75):

```
quartiles <- apply(point.cfs$Data, MARGIN = 2, FUN = quantile, probs = c(.25,.75))
ens.mean <- colMeans(point.cfs$Data)
```

```
dates <- as.POSIXlt(point.cfs$Dates$start, tz="GMT")
plot(dates, ens.mean, ylim = range(point.cfs$Data), ty = 'n', ylab = "tas - Daily Mean", xlab = "time")
polygon(x = c(dates, rev(dates)), y = c(quartiles[1, ], rev(quartiles[2, ])), border = "transparent", col = rgb(0,0,1,.4))
lines(dates, ens.mean)
```

The interface `loadECOMS` is also used in the case of gridded observations. For instance, using the same geo-location and time arguments used in the previous selection, we can now retrieve the observed temperature for that particular grid cell accessing the WFDEI dataset:

```
point.wfdei <- loadECOMS(dataset = "WFDEI", var = "tas", lonLim = -3.7, latLim = 40.4, season = 6:8, years = 2001, time =
print(object.size(point.wfdei)) # 13704 bytes
```

The WFDEI point loaded is unidimensional, corresponding to a time series of a single point location. It lacks the "member" dimension because it is an observational dataset:

```
str(point.wfdei$Data)
```

```
 num [1:92(1d)] 26.6 25.7 25.6 24.3 23.3 ...
- attr(*, "dimensions")= chr "time"
```

```
point.wfdei$xyCoords
```

```
$x
[1] -3.75

$y
[1] 40.25

attr(,"projection")
[1] "+proj=longlat +ellps=WGS84 +datum=WGS84 +no_defs +towgs84=0,0,0"
```

It is now possible to overlay the observed temperature series with the 2-month ahead prediction of CFS for that particular grid cell (the result is not very impressive, though):

```
lines(dates, point.wfdei$Data, col = "red", lwd = 1.5)
```

We could include a legend in the figure to define the different elements shown:

```
legend('bottomright', c("CFS-Members","WFDEI"), lty=c(1,1), col=c(rgb(0,0,1,.4),"red"))
```

-->