Example 3. Calculating the bias of an ensemble forecast

In this example we will calculate the bias of the multimember forecast used in the in the previous example, but this time considering the monthly aggregated data, which dramatically reduces the size of the data loaded.

First of all, we load the same data as in the previous example, but using a monthly aggregation. To this aim, first a daily aggregation is specified (`time = DD`, aggr.m = `min`), and then the aggregation function for the daily data is specified to obtain the monthly aggregation via the `aggr.m` argument:

```
ex3.cfs <- loadECOMS(dataset = "CFSv2_seasonal",
                     var = "tasmin",
                     members = 1:2,
                     lonLim = c(-15,35),
                     latLim = c(32, 75),
                     season = c(12,1,2),
                     years = 1991:2000,
                     leadMonth = 3,
                     time = "DD",
                     aggr.d = "min",
                     aggr.m = "mean")
```

Some information messages will appear on-screen indicating the steps:

```
[2015-05-15 14:18:08] Defining homogeneization parameters for variable "tasmin"
[2015-05-15 14:18:09] Defining geo-location parameters
[2015-05-15 14:18:09] Defining initialization time parameters
NOTE: Daily aggregation will be computed from 6-hourly data
NOTE: Daily data will be monthly aggregated
[2015-05-15 14:18:12] Retrieving data subset ...
[2015-05-15 14:24:31] Done
```

Note the difference in size of the daily-aggregated data of the in the previous example (35.1 Mb) as compared to the new monthly-aggregated data size (1.2 Mb):

```
print(object.size(ex3.cfs), units = "Mb")
```

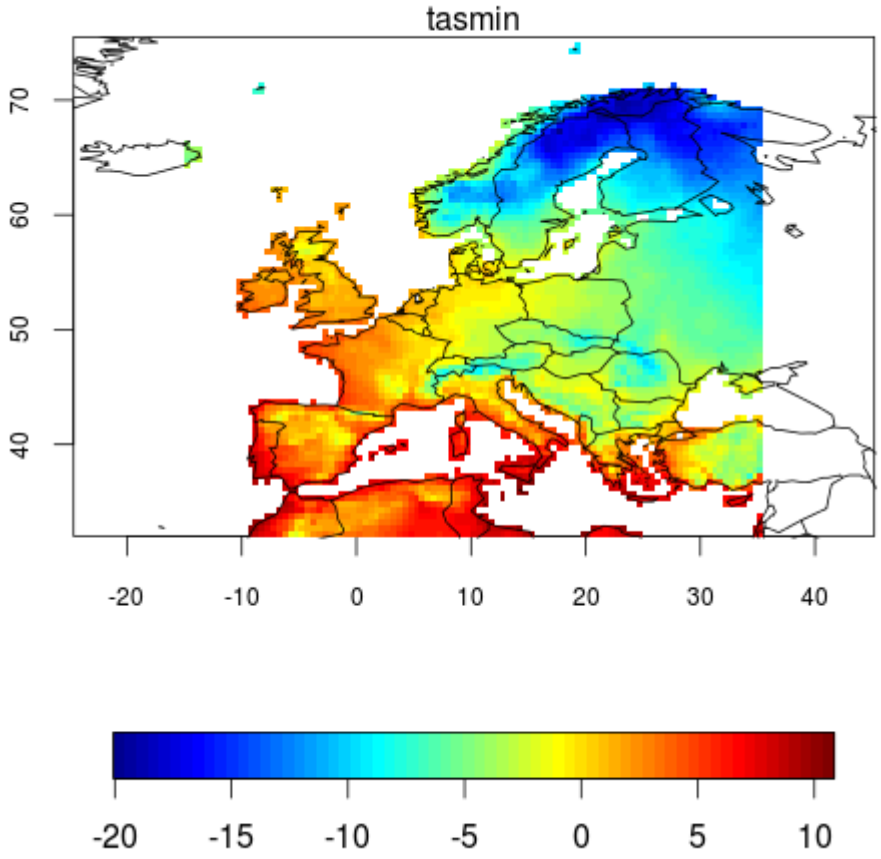
We the load the reference observations (WFDEI dataset) for the spatio-temporal domain previously chosen, using the same monthly aggregation (note that in this case the ?original data are daily, so there is no need to specify a daily aggregation):

```
ex3.obs <- loadECOMS(dataset = "WFDEI",
                     var = "tasmin",
                     lonLim = c(-15,35),
                     latLim = c(32, 75),
                     season = c(12,1,2),
                     years = 1991:2000,
                     aggr.m = "mean")
```

```
[2015-05-15 14:31:16] Defining homogeneization parameters for variable "tasmin"
[2015-05-15 14:31:16] Defining geo-location parameters
[2015-05-15 14:31:16] Defining time selection parameters
NOTE: Daily data will be monthly aggregated
[2015-05-15 14:31:17] Retrieving data subset ...
[2015-05-15 14:31:39] Done
```

This is the map of the observed mean minimum surface temperature observed for DJF 2001-2010:

```
plotMeanGrid(ex3.obs)
```



Note that WFDEI provides data for land areas only, and its spatial resolution is finer than CFS (1º vs 0.5º). In order to compare both datasets, it is first necessary to put them in the same grid (i.e., to interpolate). We use bilinear interpolation to this aim, using the `downscaleR` function `interpGridData` in combination with the `getGrid` method, useful to recover the parameters defining the grid of a dataset to pass them to the interpolator:

```
obs.regridded <- downscaleR::interpData(gridData = ex2.obs,
                                new.coordinates = getGrid(ex2),
                                method = "bilinear",
                                parallel = TRUE)
```

Note the use of the parallelization options to do the interpolation in parallel. Also, note the warnings reminding us that the extent of the input grid is wider than that from CFS. However, in this case we can safely ignore this warnings, since all the land areas we are interest in are within the CFS domain.

```
[2015-05-15 14:34:58] Performing bilinear interpolation... may take a while
[2015-05-15 14:34:58] Done
Warning messages:
1: In interpGridData(gridData = ex3.obs, new.grid = getGrid(ex3.cfs),  :
 The new longitudes are outside the data extent
2: In interpGridData(gridData = ex3.obs, new.grid = getGrid(ex3.cfs),  :
 The new latitudes are outside the data extent
```

```
plotMeanGrid(obs.regridded)
```

After regridding, both model data and observations are in the same grid. We can compute the bias. First, we calculate the mean of WFDEI, which is the reference against which to compute the biases:

```
ref <- apply(obs.regridded$Data, MARGIN = c(3,2), mean, na.rm = TRUE)
```

The following lines of code compute the bias of each member w.r.t. the reference and plot them:

```
# Now we compute the difference against each of the multimember spatial means:
require(fields)
n.members <- dim(ex3.cfs$Data)[1]
par(mfrow = c(1,2))
for (i in 1:n.members) {
     member <- apply(ex2$Data[i, , , ], MARGIN = c(3,2), mean, na.rm = TRUE)
     bias <- member - ref
     image.plot(ex2$xyCoords$x, ex2$xyCoords$y, bias, xlab = "lon", ylab = "lat", asp = 1)
     title(paste("Bias member", i))
     world(add = TRUE)
}
par(mfrow = c(1,1)) # To reset the graphical window
```