Technical Notes
Santander Meteorology Group (CSIC-UC)
`http://www.meteo.unican.es`
SMG:01.2013

The SPECS-EUPORIAS Data Portal: THREDDS Data Server and 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: joaquin.bedia@unican.es

version:v1.0–*28 Feb 2013*

Abstract

Different sector-specific impact activities to be undertaken in SPECS (`http://www.specs-fp7.eu`) and EUPORIAS (`http://www.euporias.eu`) projects require a reduced number of variables (typically at surface) from different data sources (mainly seasonal forecasts, reanalysis, and observations). The *SPECS-EUPORIAS Data Portal* has been established by the Santander Meteorology Group (UC-CSIC) as part of the data management activities in these projects to provide a unique access for these impact-relevant variables, gathered from existing datasets. This document briefly describes the current state and future plans of the data portal, which is based on a THREDDS data server providing metadata and data access using OPeNDAP and other remote data access protocols. Moreover, a user-friendly R (`http://www.r-project.org`) package has also been developed for exploring and remotely accessing subsets of data, thus reducing the burden of data access in these activities.

Data Portal URL: `http://www.meteo.unican.es/tds5/catalogs/system4/System4Datasets.html`
Documentation URL: `https://www.meteo.unican.es/trac/meteo/wiki/SpecsEuporias`

1 Introduction and Motivation

The impact activities on seasonal timescales involved in SPECS (`http://www.specs-fp7.eu`) and EUPORIAS (`http://www.euporias.eu`) projects require the use of different data sources (mainly seasonal forecasts, reanalysis, and observations). These activities include the calibration, downscaling, and modelling of sector-specific indices in agriculture, energy, health, etc., building on meteorological information. Typically, only a reduced subset of surface variables (precipitation, temperatures, mean sea level pressure, etc.) or in a reduced number of vertical levels (circulation and termodynamic drivers at, e.g., 850, 500, 200 hPa) is required for these activities. The *SPECS-EUPORIAS Data Portal* has been established by the Santander Meteorology Group (UC-CSIC) to gather the relevant information from existing datasets in order to provide a unique homogenized access to data for the SPECS and EUPORIAS partners —in particular for impact-users—.

The *SPECS-EUPORIAS Data Portal* is based on a THREDDS data server providing metadata and data access using OPeNDAP and other remote data access protocols. Moreover, since the R language (`http://www.r-project.org`) has been adopted for some key tasks in these projects —including the development of comprehensive validation and statistical-downscaling packages,— a user-friendly R package has been developed to explore and access the data portal. This package can be used in R programs to remotely access subsets of data, thus reducing the burden of data access (versions for Python and Matlab are also available under request). This package will be continuously updated (keep informed at the documentation URL above) as part of the data management activities to build a data bridge for impact users and for the R developments to be done in these projects.

This document briefly describes the current state of the data portal, which has initially focused on data from the ECMWF System4 seasonal model, as agreed in the downscaling parallel session of the kick-off meeting.

2 The THREDDS Data Server

The *SPECS-EUPORIAS Data Portal* 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 variables names, units and additional metadata follow the CF convention[1]. The variables are spatial grids based on multidimensional arrays of indexed values, following Unidata's _Coordinate convention[2,3].

Typically the data portal will include information at a daily resolution, but mothly-aggregated values could be also provided in some cases due to data limitations —in particular, Météo-France and Met Office have agreeed to provide monthly mean hindcasts for their use by the SPECS and EUPORIAS partners.— In general, the data available will be typical surface variables (e.g. precipitation and near-surface temperature), although several variables (e.g. geopotential and temperature) on pressure levels will also be stored for the statistical downscaling activities.

The data gathering activities have initially focused on the ECMWF System4 seasonal model. The Meteorological Archival and Retrieval System (MARS) is the main repository of meteorological data at the ECMWF (European Centre for Medium-Range Weather Forecasts). It contains terabytes of operational and research data as well as data from special projects[4]. The large amount of information stored and the inherent complexities of data access, download and post-processing is a first shortcoming for a flexible use of these datasets by a large number of partners. To overcome this issue, a reduced subset of surface variables[5] (precipitation, temperatures and mean sea level pressure) have been downloaded from MARS (a colection of GRIB-1 files) at 0.75° spatial resolution and made available throught the *SPECS-EUPORIAS data portal*. The downloaded data has been exposed as three different virtual datasets using THREDDS:

- **System4 seasonal range (15 members)**: There are twelve initializations (hereafter called "runtimes") per year (the first of January, February, ...) running for 7 months (hereafter called simply "times"). An ensemble of 15 members is available for the whole 1981-2010 period.
- **System4 seasonal range (51 members)**: There are only four runtimes per year (the first of February, May, August and November) and the forecasts run for 7 months. An ensemble of 51 members is available for the whole 1981-2010 period.
- **System4 annual range (15 members)**: As in the previous case, there are four runtimes per year, but the forecasts run for 13 months. An ensemble of 15 members is available for the whole 1981-2010 period.

Data gathering activities will next move to the CFS (`http://cfs.ncep.noaa.gov`) version 2 hindcast, developed at the Environmental Modeling Center at NCEP and also to reanalysis and observational datasets.

Although the THREDDS server provides a web interface to explore and access the datasets (shown in Sec. 4), it is strongly recommented the use of OPeNDAP (DODS) client libraries to remotely access the data from scientific computing environments (`R`, Matlab, Python, etc.). For instance, the `R` function provided in this tutorial is based on the NetCDF Java DODS client[6], using the `rJava` `R` package (a similar approach has been also followed for the Matlab implementation). Alternatively, the most recent NetCDF library versions provide access to OPenDAP datasets (this has been the choice for the Python implementation). In the following, we show a simple example of data access using the `R` package developed as part of the data portal. In particular the System4 datasets can by directly accessed using the `loadSystem4` function, allowing the retrieval of slices for a particular variable in any of the dataset dimensions (member/space/runtime/time). Note that a more ellaborated worked example using `R` is shown in Appendix A. Moreover, for a better understanding of the datasets structure, the use of the web interface for the OPeNDAP service is also illustrated Sec. 4.

[1] `http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.4/cf-conventions.html`

[2] `http://www.unidata.ucar.edu/software/netcdf-java/reference/CoordinateAttributes.html`

[3] `http://www.unidata.ucar.edu/software/netcdf-java/tutorial/GridDatatype.html`

[4] `http://www.ecmwf.int/services/archive/`

[5] `http://www.ecmwf.int/products/changes/system4/technical_description.html#description`

[6] http://www.unidata.ucar.edu/software/netcdf-java/documentation.htm

3 Accesing the Data Portal via R

The *SPECS-EUPORIAS Data Portal* can be remotely accessed from R programs using the loadSystem4 function available at the **documentation URL** given in the abstract. Note that this function is part of a comprehensive R package currently under development. This function automatically cares about the proper location of the right indices for data sub-setting across the different variable dimensions, given a few simple arguments for subset definition. In addition, instead of retrieving a NetCDF file that needs to be opened and read, the requested data is directly loaded into the current R working session, according to a particular structure described below.

Two preliminary steps are required before starting to use the loadSystem4 function:

1. The Java client for DODS need to be present in the current working directory. This corresponds to the netcdfAll-4.3.jar file. When sourcing the loadSystem4.R file, the existence of this file will be checked for, and if absent, it will be automatically downloaded.
2. Before accessing the data, authentication is required at the *SPECS-EUPORIAS Data Portal.* This can be done directly from R with the following calls to rJava commands[7]:

```
> source("loadSystem4.R")
> username <- "myUsername"
> password <- "myPassword"
> aux <- .jnew("ucar/nc2/util/net/HTTPBasicProvider", username, password)
> J("ucar.nc2.util.net.HTTPSession")$setGlobalCredentialsProvider(aux)
```

Once these two requirements are fulfilled, the function can be used for the rest of the session. In the next lines we describe an illustrative example considering one-month lead time forecasts of minimum surface temperature for January over a window centered in Europe —0°W – 30°E and 35°S – 65°N—. A more ellaborated example describing a multi-model selection of a similar dataset will be further developed in Appendix A.

```
> ds <- "http://www.meteo.unican.es/tds5/dodsC/system4/System4_Seasonal_15Members.ncml";
> openDAP.query <- loadSystem4(dataset = ds, var = "tasmin", members = 1,
+       lonLim = c(0,30), latLim = c(35,65),
+       season = 1, years = 1981:2000, leadMonth = 1)
```

where the function arguments are the following:

- dataset: A character string indicating the full URL path to the OPeNDAP dataset. Currently, the accepted values correspond to the System4 datasets described in Sec. 2, as specified in the above example, but using System4_Seasonal_15Members.ncml, System4_Seasonal_51Members.ncml, or System4_Annual_15Members.ncml ending strings.
- var: Variable code. Currently tas, tasmin, tasmax, pr, mslp, as internally defined in the dictionary defined for System4 following the nomenclature displayed in Table 1, although new variables and datasets will be soon included.
- members: Optional. Default to all members. In the above case, a single member (the first) of the System4 ensemble is loaded, but additional members could be also specified (e.g. members=NULL for all members, or members=1:5 for the first five members).
- lonLim: Vector of length = 2, with minimum and maximum longitude coordinates, in decimal degrees, of the bounding box selected.
- latLim: Vector of length = 2, with minimum and maximum latitude coordinates, in decimal degrees, of the bounding box selected.
- season: A vector of integers specifying the desired "season" (in months, January=1, etc.) of analysis. Options include a single month (as in the above example) or a standard season (e.g. period = c(12,1,2) for standard Boreal winter, DJF).
- years: Optional. Default to all available years. Vector of years to select. Note that in cases with year-crossing seasons (e.g. winter DJF, season = c(12,1,2), for a particular year period years = 1981:2000), by convention the first season would be DJF 1980/81, if available (otherwise a warning

[7] For further details on this step see http://www.unidata.ucar.edu/software/netcdf-java/v4.3/javadocAll/ucar/nc2/util/net/HTTPBasicProvider.html

message is given).

- `loadMonth`: Lead month forecast time corresponding to the first month of the specified season. Note that `leadMonth = 1` for `season = 1` (January) correspond to the December initilization forecasts. In this way the effect of the lead time forecast in the analysis of a particular season can be analyzed by just changing this parameter.

The result of the function is a data structure with all the requested information as follows.

```
> str(openDAP.query)
List of 4
 $ MemberData   :List of 1
  ..$ : num [1:930, 1:1600] 275 277 278 279 277 ...
 $ Coordinates  : num [1:1600, 1:2] 64.5 63.7 63 62.2 61.5 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:2] "lat" "lon"
 $ RunDates     : POSIXlt[1:30], format: "1981-12-01" "1982-12-01" "1983-12-01" ...
 $ ForecastDates: Date[1:930], format: "1982-01-01" "1982-01-02" "1982-01-03" ...
```

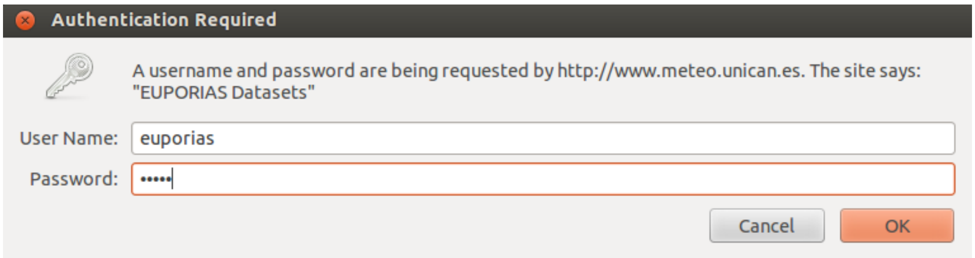
- `MemberData`: This is a list of length n, where n = number of members of the ensemble selected in the `member` argument. Each element of the dataset is a 2-D matrix of i rows × j columns, of i forecast times and j grid-points
- `Coordinates`: A 2-D matrix of j rows (where j = number of grid points selected) and two columns corresponding to the latitude and longitude coordinates, in this order.
- `RunDates`: A `POSIXlt` time object corresponding to the initialization times selected.
- `ForecastDates`: A vector of class `Date` of length i, corresponding to the rows of each matrix in `MemberData`, containing the verification dates.

Table 1: Values of argument `var` (vocabulary) allowed by function `loadSystem4` and their correspondence with the actual names of the variable as coded in the System4 database (dictionary for System4). Note that System4 precipitation is internally transformed to correspond to daily accumulated values.

Short Name	Dataset variable
`tasmax`	Maximum_temperature_at_2_metres_since_last_24_hours_surface
`tasmin`	Minimum_temperature_at_2_metres_since_last_24_hours_surface
`tas`	Mean_temperature_at_2_metres_since_last_24_hours_surface
`pr`	Total_precipitation_surface
`mslp`	Mean_sea_level_pressure_surface

4 Accesing the Data Portal via Web

The *SPECS-EUPORIAS Data Portal* can be accessed through the **Data Portal URL** provided in the abstract. First of all, an authentication dialog will request a valid user name and password.



Authentication Required

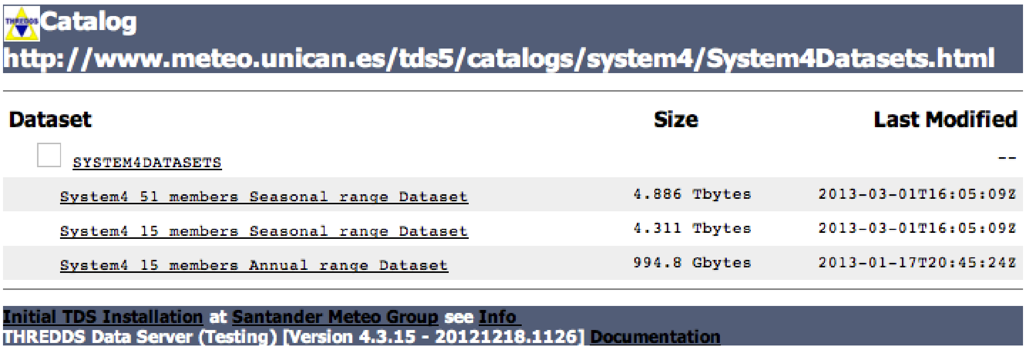
A username and password are being requested by http://www.meteo.unican.es. The site says: "EUPORIAS Datasets"

User Name: euporias

Password: •••••

Figure 1: Authentication dialog

Afterwards, the different datasets described in Sec. 2 are listed as links in the web browser window (Fig. 2). By clicking in any of the datasets, a new window will appear providing information on the variables and geospatial and time coverages, and offering different options for data access and/or visualization (Fig. 3). Currently, only the OPeNDAP access service is fully operative in the portal. Therefore, in this example, we will illustrate the use of this service, which allows selecting time/spatial data slices from the OPeNDAP data access form shown in Fig. 4 and downloading the resulting data in both ASCII and Binary formats.

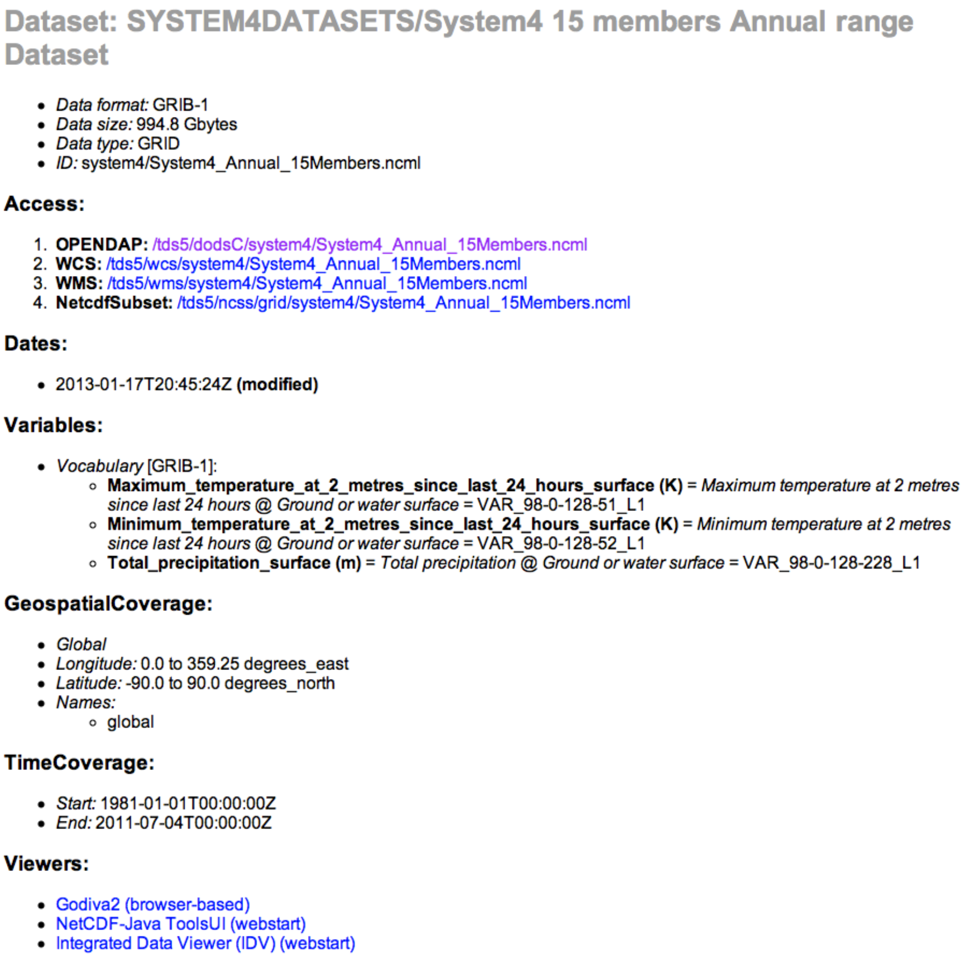


Catalog
http://www.meteo.unican.es/tds5/catalogs/system4/System4Datasets.html

Dataset	Size	Last Modified
□ SYSTEM4DATASETS		--
System4 51 members Seasonal range Dataset	4.886 Tbytes	2013-03-01T16:05:09Z
System4 15 members Seasonal range Dataset	4.311 Tbytes	2013-03-01T16:05:09Z
System4 15 members Annual range Dataset	994.8 Gbytes	2013-01-17T20:45:24Z

Initial TDS Installation at Santander Meteo Group see Info
THREDDS Data Server (Testing) [Version 4.3.15 - 20121218.1126] Documentation

Figure 2: Catalog of the EUPORIAS-SPECS System4 datasets. Note that although they only include a few variables, their size range from one to four Terabytes.



Dataset: SYSTEM4DATASETS/System4 15 members Annual range Dataset

- *Data format:* GRIB-1
- *Data size:* 994.8 Gbytes
- *Data type:* GRID
- *ID:* system4/System4_Annual_15Members.ncml

Access:

1. **OPENDAP:** /tds5/dodsC/system4/System4_Annual_15Members.ncml
2. **WCS:** /tds5/wcs/system4/System4_Annual_15Members.ncml
3. **WMS:** /tds5/wms/system4/System4_Annual_15Members.ncml
4. **NetcdfSubset:** /tds5/ncss/grid/system4/System4_Annual_15Members.ncml

Dates:

- 2013-01-17T20:45:24Z **(modified)**

Variables:

- *Vocabulary* [GRIB-1]:
  - **Maximum_temperature_at_2_metres_since_last_24_hours_surface (K)** = *Maximum temperature at 2 metres since last 24 hours @ Ground or water surface* = VAR_98-0-128-51_L1
  - **Minimum_temperature_at_2_metres_since_last_24_hours_surface (K)** = *Minimum temperature at 2 metres since last 24 hours @ Ground or water surface* = VAR_98-0-128-52_L1
  - **Total_precipitation_surface (m)** = *Total precipitation @ Ground or water surface* = VAR_98-0-128-228_L1

GeospatialCoverage:

- *Global*
- *Longitude:* 0.0 to 359.25 degrees_east
- *Latitude:* -90.0 to 90.0 degrees_north
- *Names:*
  - global

TimeCoverage:

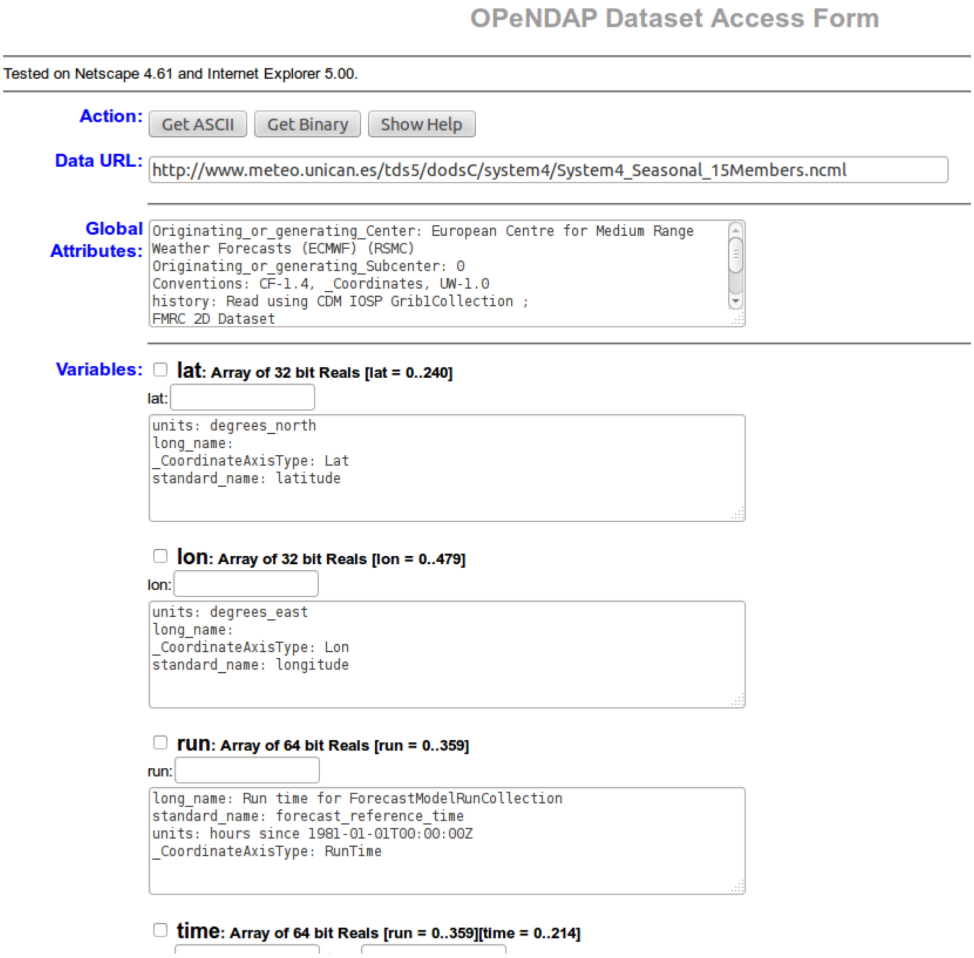
- *Start:* 1981-01-01T00:00:00Z
- *End:* 2011-07-04T00:00:00Z

Viewers:

- Godiva2 (browser-based)
- NetCDF-Java ToolsUI (webstart)
- Integrated Data Viewer (IDV) (webstart)

Figure 3: Detail of a particular dataset with information on the included variables and geospatial and time coverages. The different options for data access and visualization are also shown.

Note that, as explained before, the variables provided by the data portal (e.g. minimum temperature) are stored as gridsets. Thus, in addition to these variables, also auxiliary coordinate variables (lat, lon, run, time, member) should be handled for geo-temporal data referencing (see Fig. 4). Moreover, three time coordinates are included as referece for different grid variables because they are defined for different forecast times (one extra time for precipitation and different temporal resolution for mean sea level pressure). Note that this highly complicates the direct analysis of the data and, hence, this options is only recommend for data exploration. In the following we show how to use this service to explore the structure of the datasets and to obtain simple pieces of information in ASCII format.



OPeNDAP Dataset Access Form

Tested on Netscape 4.61 and Internet Explorer 5.00.

Action: Get ASCII | Get Binary | Show Help

Data URL: http://www.meteo.unican.es/tds5/dodsC/system4/System4_Seasonal_15Members.ncml

Global Attributes:

```
Originating_or_generating_Center: European Centre for Medium Range
Weather Forecasts (ECMWF) (RSMC)
Originating_or_generating_Subcenter: 0
Conventions: CF-1.4, _Coordinates, UW-1.0
history: Read using CDM IOSP GriblCollection ;
FMRC 2D Dataset
```

Variables:

□ **lat**: Array of 32 bit Reals [lat = 0..240]

lat:

```
units: degrees_north
long_name:
_CoordinateAxisType: Lat
standard_name: latitude
```

□ **lon**: Array of 32 bit Reals [lon = 0..479]

lon:

```
units: degrees_east
long_name:
_CoordinateAxisType: Lon
standard_name: longitude
```

□ **run**: Array of 64 bit Reals [run = 0..359]

run:

```
long_name: Run time for ForecastModelRunCollection
standard_name: forecast_reference_time
units: hours since 1981-01-01T00:00:00Z
_CoordinateAxisType: RunTime
```

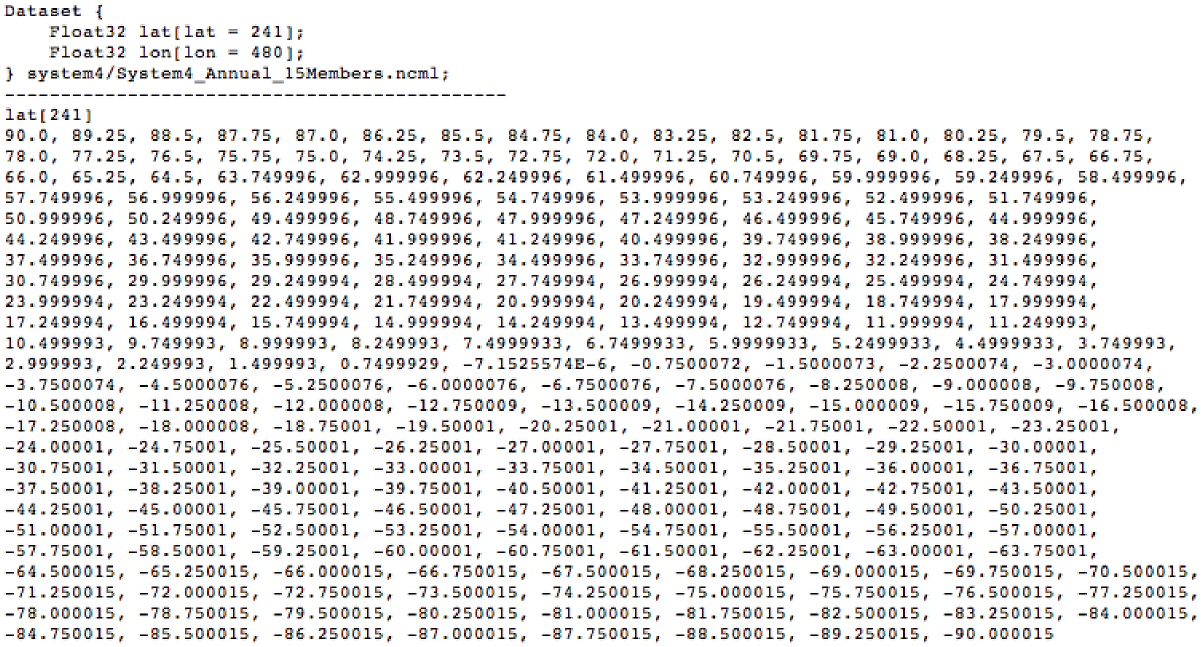
□ **time**: Array of 64 bit Reals [run = 0..359][time = 0..214]

Figure 4: Detail of the OPeNDAP dataset access form for a particular dataset.

By default, if no specifications are given in the different subsetting boxes of the OpenDAP form, the whole data on the whole spatio/temporal and member ranges of the dataset would be accessed. However, this option will raise an error due to the large size of the request (the maximum size of a single request has been set to 100 Mbytes in the `SPECS-EUPORIAS data portal` for the sake of multi-connection efficiency). The basic steps to retrieve subsets of data are the following:

- To select a variable click on the checkbox to its left.
- To constrain the variable, edit the information that appears in the text boxes below the variable. This is a vector of integers indicating index positions of length three, with the following order: `[start:stride:end]`.
- To get ASCII or binary values for the selected variables, click on the Get ASCII or Get Binary buttons of the Action field. Note that the URL displayed in the Data URL field is updated as you select and/or constrain variables. The URL in this field can be cut and pasted in various DODS clients.

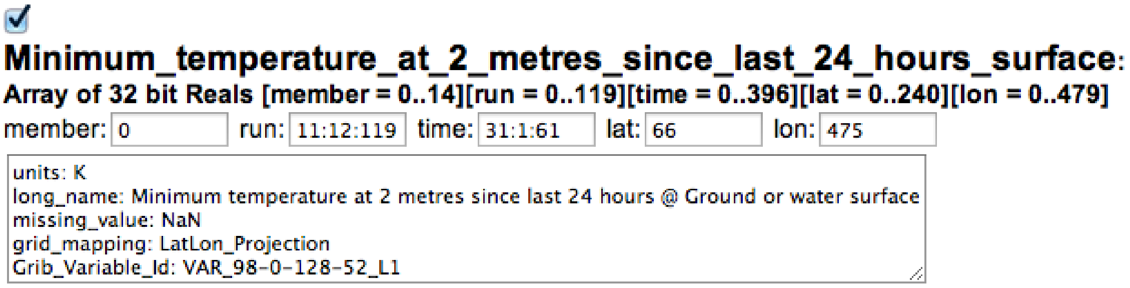
The main disadvantage of the OpenDAP service from the end-user point of view is that the specifications for subsetting dimensions are not given in their original magnitudes (i.e., latitudes and longitudes are not given in decimal degrees), but by the indexes of their position along their respective axes (note that first index value is always 0). Thus, to find out the indexes for the desired selection, we need to dump and analyze the particular values defined in the coordinate variable. For instance, Fig. 5 shows the 241 values defined for the `lat` (latitude) coordinate, as provided by the Get ASCII option (checking the corresponding check-box).



```
Dataset {
    Float32 lat[lat = 241];
    Float32 lon[lon = 480];
} system4/System4_Annual_15Members.ncml;
---------------------------------------------
lat[241]
90.0, 89.25, 88.5, 87.75, 87.0, 86.25, 85.5, 84.75, 84.0, 83.25, 82.5, 81.75, 81.0, 80.25, 79.5, 78.75,
78.0, 77.25, 76.5, 75.75, 75.0, 74.25, 73.5, 72.75, 72.0, 71.25, 70.5, 69.75, 69.0, 68.25, 67.5, 66.75,
66.0, 65.25, 64.5, 63.749996, 62.999996, 62.249996, 61.499996, 60.749996, 59.999996, 59.249996, 58.499996,
57.749996, 56.999996, 56.249996, 55.499996, 54.749996, 53.999996, 53.249996, 52.499996, 51.749996,
50.999996, 50.249996, 49.499996, 48.749996, 47.999996, 47.249996, 46.499996, 45.749996, 44.999996,
44.249996, 43.499996, 42.749996, 41.999996, 41.249996, 40.499996, 39.749996, 38.999996, 38.249996,
37.499996, 36.749996, 35.999996, 35.249996, 34.499996, 33.749996, 32.999996, 32.249996, 31.499996,
30.749996, 29.999996, 29.249994, 28.499994, 27.749994, 26.999994, 26.249994, 25.499994, 24.749994,
23.999994, 23.249994, 22.499994, 21.749994, 20.999994, 20.249994, 19.499994, 18.749994, 17.999994,
17.249994, 16.499994, 15.749994, 14.999994, 14.249994, 13.499994, 12.749994, 11.999994, 11.249993,
10.499993, 9.749993, 8.999993, 8.249993, 7.4999933, 6.7499933, 5.9999933, 5.2499933, 4.4999933, 3.749993,
2.999993, 2.249993, 1.499993, 0.7499929, -7.1525574E-6, -0.7500072, -1.5000073, -2.2500074, -3.0000074,
-3.7500074, -4.5000076, -5.2500076, -6.0000076, -6.7500076, -7.5000076, -8.250008, -9.000008, -9.750008,
-10.500008, -11.250008, -12.000008, -12.750009, -13.500009, -14.250009, -15.000009, -15.750009, -16.500008,
-17.250008, -18.000008, -18.75001, -19.50001, -20.25001, -21.00001, -21.75001, -22.50001, -23.25001,
-24.00001, -24.75001, -25.50001, -26.25001, -27.00001, -27.75001, -28.50001, -29.25001, -30.00001,
-30.75001, -31.50001, -32.25001, -33.00001, -33.75001, -34.50001, -35.25001, -36.00001, -36.75001,
-37.50001, -38.25001, -39.00001, -39.75001, -40.50001, -41.25001, -42.00001, -42.75001, -43.50001,
-44.25001, -45.00001, -45.75001, -46.50001, -47.25001, -48.00001, -48.75001, -49.50001, -50.25001,
-51.00001, -51.75001, -52.50001, -53.25001, -54.00001, -54.75001, -55.50001, -56.25001, -57.00001,
-57.75001, -58.50001, -59.25001, -60.00001, -60.75001, -61.50001, -62.25001, -63.00001, -63.75001,
-64.500015, -65.250015, -66.000015, -66.750015, -67.500015, -68.250015, -69.000015, -69.750015, -70.500015,
-71.250015, -72.000015, -72.750015, -73.500015, -74.250015, -75.000015, -75.750015, -76.500015, -77.250015,
-78.000015, -78.750015, -79.500015, -80.250015, -81.000015, -81.750015, -82.500015, -83.250015, -84.000015,
-84.750015, -85.500015, -86.250015, -87.000015, -87.750015, -88.500015, -89.250015, -90.000015
```

Figure 5: Text file displaying the values for the `lat` (latitude) coordinate variable.

Using these facilities it can be obtained after some calculations that the closest `lat` and `lon` coordinates for a particular location of interest (e.g. Madrid) are 66 and 475, respectively. Thus, the time series for Madrid corresponding to the example described in the previous section (minimum temperature forecasts for January with one-month lead time, i.e. from the simulations started the first of December) could be requested as shown in Fig. 6. Note that the indices selected for the run coordinate correspond to the December initilizations (index positions 11, 23,...; note that indexes start in 0) and for the time coordinate correspond to January (positions, 31 to 62, in days after the run time). Note that the proper use of this service requires a full understanding of the data structure and, therefore, it is only advised for data exploration.



☑ **Minimum_temperature_at_2_metres_since_last_24_hours_surface:**
Array of 32 bit Reals [member = 0..14][run = 0..119][time = 0..396][lat = 0..240][lon = 0..479]
member: 0 run: 11:12:119 time: 31:1:61 lat: 66 lon: 475

units: K
long_name: Minimum temperature at 2 metres since last 24 hours @ Ground or water surface
missing_value: NaN
grid_mapping: LatLon_Projection
Grib_Variable_Id: VAR_98-0-128-52_L1

Figure 6: Detail of the query from the OPeNDAP dataset access form to retrieve a subset (a time series for a single gridbox) of minimum temperature.

Appendix

A Example of Data Analysis with R

The first step to get things started is loading the rJava library[8], which provides a low-level R-to-java interface, allowing the call to methods and accessing fields of the netcdf-java version 4.3.15 (see http://www.unidata.ucar.edu/software/netcdf-java/v4.3/javadocAll/overview-summary.html for an overview). The rationale behind the use of the Java methods and classes within R is to take advantage of the efficiency of the object-oriented philosophy of Java, and in particular the wide range of possibilities for netCDF manipulation, in combination with the powerful statistical and graphical capabilities of the R environment and its efficient handling of vectors, arrays and other matrix-like objects.

When sourcing the loadSystem4.R file, in addition to loading the function and the required rJava package, the path to the .jar file is specified in order to initialize the Java Virtual Machine, using the .jinit command. If the .jar file is not available, it will be automatically dowloaded from ftp://ftp.unidata.ucar.edu/pub/netcdf-java/v4.3/netcdfAll-4.3.jar.

```
> source("loadSystem4.R")
```

Note that after sourcing the file, and prior to the use of loadSystem4 function, we must provide credentials for accessing the THREDDS data server, as previously described in Section 3. In the next lines we show an example of how to access and analyze a 15 member subset from R. We will select a domain covering part of Europe (approximately 2000 grid points) and will retrieve the data corresponding to the seasonal forecast of minimum surface (2m) temperature of January for the period 1991-2000, considering a lead month of 2 (i.e., run times starting in November of the previous year). In this particular example, we will ask for the data of all available ensemble members (n = 15). In addition, we will show time series for selected points. In this example, we have chosen the locations of several European capitals.

```
> cities <- matrix(c(48.83, 2.37, 40.40, -3.72, 53.34, -6.25, 59.92, 10.76, 52.24, 21.03,
+       41.89, 12.49, 37.98, 23.72, 52.52, 13.40, 60.17, 24.94), ncol = 2, byrow = TRUE)
> colnames(cities) <- c("lat","lon")
> rownames(cities) <- c("Paris","Madrid","Dublin","Oslo","Warsaw", "Rome", "Athens",
+                          "Berlin", "Helsinki")
> print(cities)
```

```
           lat   lon
Paris    48.83  2.37
Madrid   40.40 -3.72
Dublin   53.34 -6.25
Oslo     59.92 10.76
Warsaw   52.24 21.03
Rome     41.89 12.49
Athens   37.98 23.72
Berlin   52.52 13.40
Helsinki 60.17 24.94
```

The size of the requested data will be of 15 matrices of approximately 2000 columns (2120) by 31 × 10 forecast times arranged in rows. This request would be formulated as follows:

```
> ds = "http://www.meteo.unican.es/tds5/dodsC/system4/System4_Seasonal_15Members.ncml"
> mn = loadSystem4(dataset = ds, var = "tasmin", lonLim = c(-10,30), latLim = c(35,65),
+            season = 1, years = 1991:2000, leadMonth = 2)
```

Note that the members argument is missing. Thus, by default, the function will load all members.

[8] Simon Urbanek (2011). *rJava: Low-level R to Java interface.* R package version 0.9-3. http://CRAN.R-project.org/package=rJava

The above data access step took 555 seconds (about 9 minutes) with a wired broadband connection.

The structure of the data returned has been conceived to minimize the amount of accesory information. Thus, only the strictly necessary information for data interpretation, analysis and plotting is returned. Future versions may contain new features on demand of the users. Needless to say, the user can also modify the function's code at his/her convenience, in order to retrieve additional information (e.g. projections, units . . .). Next, a summary of the resulting output is displayed:

```
> str(mn)
```

```
List of 4
 $ MemberData   :List of 15
  ..$ : num [1:310, 1:2120] 278 277 278 277 277 ...
  ..$ : num [1:310, 1:2120] 280 279 276 277 273 ...
  ..$ : num [1:310, 1:2120] 277 277 279 280 280 ...
  ..$ : num [1:310, 1:2120] 275 275 277 279 277 ...
  ..$ : num [1:310, 1:2120] 276 274 276 275 276 ...
  ..$ : num [1:310, 1:2120] 271 271 276 278 278 ...
  ..$ : num [1:310, 1:2120] 279 278 277 277 275 ...
  ..$ : num [1:310, 1:2120] 276 276 277 275 275 ...
  ..$ : num [1:310, 1:2120] 276 276 279 277 276 ...
  ..$ : num [1:310, 1:2120] 276 276 277 277 277 ...
  ..$ : num [1:310, 1:2120] 272 272 276 275 277 ...
  ..$ : num [1:310, 1:2120] 275 272 272 271 274 ...
  ..$ : num [1:310, 1:2120] 270 270 277 275 275 ...
  ..$ : num [1:310, 1:2120] 280 277 277 279 278 ...
  ..$ : num [1:310, 1:2120] 272 272 271 271 269 ...
 $ Coordinates  : num [1:2120, 1:2] 64.5 63.7 63 62.2 61.5 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:2] "lat" "lon"
 $ RunDates     : POSIXlt[1:10], format: "1990-11-01" ...
 $ ForecastDates: Date[1:310], format: "1991-01-01" ...
```

In the following lines of code we will compute the multi-member ensemble mean of minimum temperatures of January of the domain selected. Note that we also apply the conversion from °K to °C.

```
> aux = matrix(NA, nrow = nrow(mn$Coordinates), ncol = length(mn$MemberData))
> for (i in 1:length(mn$MemberData)) {
+       aux[ ,i] = colMeans(mn$MemberData[[i]]) - 273.15
+ }
> mean.field = rowMeans(aux)
```

The `mean.field` object is a vector of length equal to the number of rows of the coordinates matrix. For each grid point, it gives the mean minimum temperature for January of the multi-member ensemble. Next, we illustrate a simple procedure to plot the data. Note that specific plot methods for the `loadSystem4` function are currently under development. In this example, we make use of quite standard R procedures and libraries for plotting spatial data.

Because of the projection of the data, the grid points in the latitude axis are not exactly regularly spaced. In this example we use the bivariate interpolation procedure implemented in the R library `akima`[9], while preserving the native resolution of 0.75° for data representation in the regular space.

```
> library(akima)
> x = mn$Coordinates[ ,2]
> y = mn$Coordinates[ ,1]
> z = interp(x, y, mean.field, xo = seq(min(x), max(x), .75), yo = seq(min(y), max(y), .75))
> image(z, asp = 1, col = topo.colors(20))
```

[9] Fortran code by H. Akima R port by Albrecht Gebhardt (2013). *akima: Interpolation of irregularly spaced data.* R package version 0.5-9. `http://CRAN.R-project.org/package=akima`

```
> contour(z, levels = pretty(z$z, 20), add = TRUE)
> points(cities[ ,2:1], pch = 15, col = "red", cex = 1.2)
```

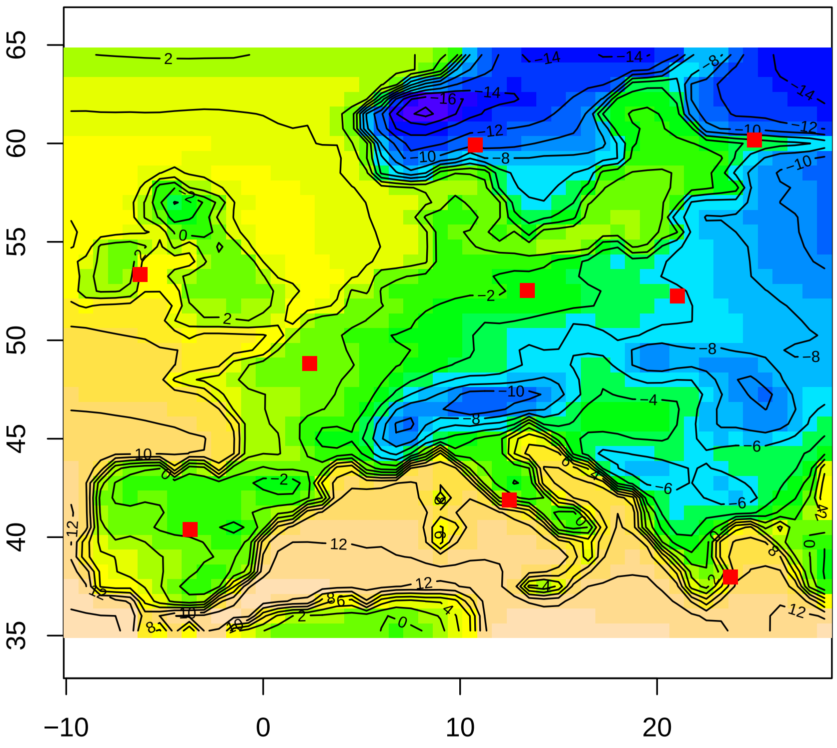


Figure 7: Hindcast mean of minimum surface temperature in January, for the period 1991-2000. Hindcast initialized in November. Mean values shown correspond to the multi-member ensemble considering all 15 members. The red squares represent the selected points contained in matrix `cities`.

In the next lines we show an example of time series obtained from the data previously presented, at the different european cities displayed in the previous figure. The goal in this case is to show both the mean and the spread of the different members at each particular location. Note that we load the R package `sp`[10], used to compute the euclidean distances needed for the calculation of the nearest grid cells.

```
> library(sp) # Required to compute distances
> # This part is used to build an index of the positions of the labels in the X axis
> aux = as.POSIXlt(mn$ForecastDates)$year
> index <- c()
> for (i in 1:(length(aux) - 1)) {
+       if (aux[i + 1] - aux[i] > 0) {
+       index <- c(index,i)
+       }
+ }
> x.lab.pos = c(1, index)
> # In the next lines the mean and ranges at each point are computed and plotted
> par(mfrow = c(3,3))
> for (i in 1:nrow(cities)) {
```

[10] Pebesma, E.J. and R.S. Bivand (2005). *Classes and methods for spatial data in R.* R News 5(2). `http://cran.r-project.org/doc/Rnews/`.

```
+       aux = matrix(NA, ncol = length(mn$MemberData), nrow = nrow(mn$MemberData[[1]]))
+       index = which.min(spDistsN1(mn$Coordinates, cities[i, ]))
+       for (j in 1:length(mn$MemberData)) {
+             aux[ ,j] = mn$MemberData[[j]][ ,index] - 273.15
+       }
+       mean.series = rowMeans(aux)
+       max.series = apply(aux, MAR = 1, FUN = max)
+       min.series = apply(aux, MAR = 1, FUN = min)
+       monthly.means = rep(NA, length(x.lab.pos))
+       ref.rows = c(x.lab.pos, length(mean.series))
+       for (k in 1:(length(ref.rows)-1)) {
+            monthly.means[k] = mean(aux[ref.rows[k] : (ref.rows[k+1]-1), ])
+       }
+       plot(mean.series, ty='l', axes=FALSE, xlab="", ylab="tasmin",
+            ylim = c(-30,15))
+       polygon(c(1 : length(mean.series), length(mean.series) : 1),
+               c(max.series, rev(min.series)),
+               col="grey", border = "transparent")
+       lines(mean.series, col = "blue")
+       monthly.means = rep(NA, length(x.lab.pos))
+       ref.rows = c(x.lab.pos, length(mean.series))
+       for (k in 1:(length(ref.rows)-1)) {
+            monthly.means[k] = mean(aux[ref.rows[k] : (ref.rows[k+1]-1), ])
+       }
+       abline(h = mean(mean.series), lty=2)
+       points(x.lab.pos + 15.5, monthly.means, pch = 4, col = "red")
+
+       axis(2)
+       axis(1, labels = unique(as.POSIXlt(mn$ForecastDates)$year + 1900),
+            at = x.lab.pos)
+       abline(v = x.lab.pos, lty = 3)
+       title(main = paste(rownames(cities)[i]), cex.main = 1.2)
+       mtext(paste(cities[i,1],",",cities[i,2]))
+ }
```

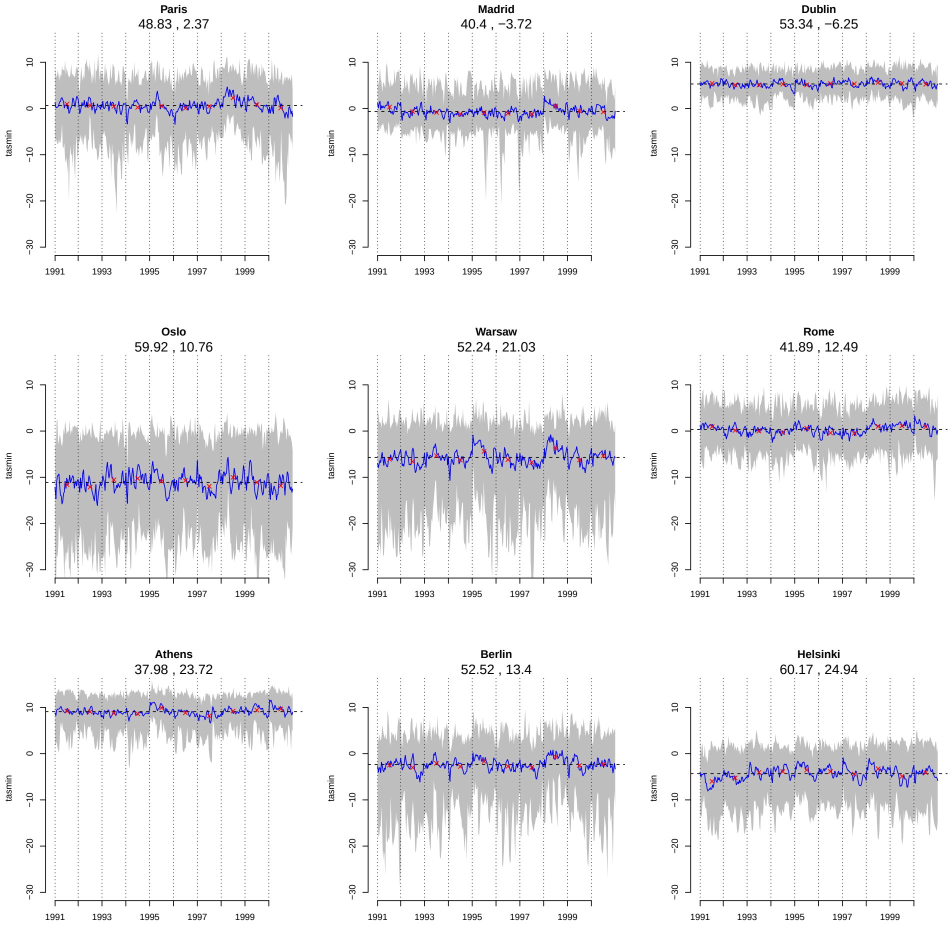


Figure 8: Hindcast minimum surface temperature of January (initialized in November, period 1991-2000) at the european capital contained in matrix `cities`. The blue line corresponds to the mean of the multi-member ensemble (n = 15). The shadowed area indicates the minimum and maximum range of all members. Red crosses correspond to the monthly mean values. The horizontal dashed black line indicates the mean of the whole period (1991-2000).