I

SIG y Geografía

SÉMATA, *Ciencias Sociais e Humanidades,* 2015, vol. 27: 11-37

El uso de bases de datos climáticos netCDF con estructura matricial en el entorno de R

A practical introduction in the use of netCDF in the environment of R

DOMINIC ROYÉ
Universidad de Cantabria

Resumen

La información espacio-temporal es en la actualidad un elemento clave en disciplinas como la Climatología y la Meteorología. Un formato de uso muy extendido es el de las bases de datos netCDF, que permiten obtener una estructura multidimensional e intercambiar los datos de forma independiente al sistema operativo empleado. En este artículo se introduce el uso de estas bases de datos con el entorno de software libre R. Para ello se utiliza una cuadrícula de la temperatura máxima de la Península Ibérica para el período 1971-2007. El objetivo es poder leer y visualizar el formato netCDF realizando ejemplos de cálculos globales y otros más específicos. Finalmente se muestra la aplicabilidad en un caso de estudio: la amplitud diurna en la Península Ibérica para los meses de enero y agosto 2006.
Palabras clave: netCDF, R, climatología, temperatura, matriz, base de datos

Abstract

Spatio-temporal data is currently key to many disciplines, especially to climatology and meteorology. A widespread format is netCDF allowing a multidimensional structure and an exchange of data machine independently. In this article, we introduce the use of these databases with the free software environment R. To do this, we will work with a grid of the maximum temperature of the Iberian Peninsula for the period 1971-2007. The goal is to read and visualize the netCDF format, and make some first overall and specific calculations. Finally the applicability is shown in a case study: the diurnal temperature variation in the Iberian Peninsula for January and August 2006.
Keywords: netCDF, R, climatology, temperature, matrix, database

1. INTRODUCCIÓN

La información espacio-temporal es clave en muchas disciplinas, especialmente en la Climatología o la Meteorología, y ello hace necesario disponer de un formato que

permita una estructura multidimensional. Además es importante que ese formato tenga un alto grado de compatibilidad de intercambio y pueda almacenar un elevado número de datos. Estas características llevaron al desarrollo del estándar abierto netCDF (*Network Common Data Form*). Existen múltiples programas para el tratamiento o la visualización de netCDFs (por ejemplo, Matlab, ArcGIS o Python). En este trabajo se hace uso de netCDF mediante el entorno de software libre de R por las numerosas y diversas técnicas estadísticas, la manipulación de datos y la elaboración de salidas gráficas.

1.1. NetCDF

El formato netCDF es un estándar abierto de intercambio de datos científicos multidimensionales que se utiliza con datos de observaciones o modelos, principalmente en disciplinas como la Climatología, la Meteorología y la Oceanografía. La convención netCDF es gestionada por Unidata (unidata.ucar.edu/software/netcdf). Se trata de un formato espacio-temporal con una cuadrícula regular o irregular. Según la página de UniData: "*NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data*". La estructura multidimensional en forma de matriz (*array*) permite usar no sólo datos espacio-temporales sino también multivariables. Un ejemplo de tres dimensiones podría constituir la temperatura de aire en la Península Ibérica para un periodo de 30 años, visualizado esquemáticamente en la Figura 1. Se obtendría una cuarta dimensión al

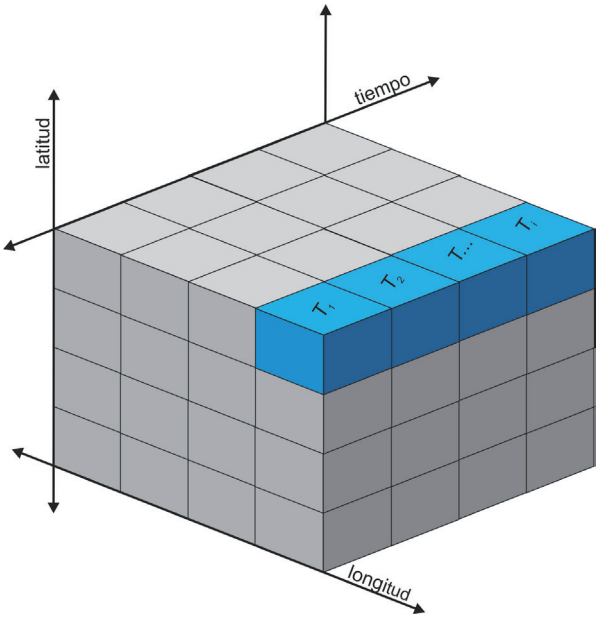


Figura 1. Ejemplo de una matriz tridimensional del formato netCDF.

añadir a este ejemplo otra variable como la altura geopotencial. Las características generales del netCDF se refieren al uso de un sistema de coordenadas n-dimensional, de múltiples variables y de una rejilla regular o irregular. Además se incluyen metadatos que describen los contenidos. La extensión del formato netCDF es habitualmente "nc". Es importante indicar que el formato hace uso de la *CF Convention* (*Climate and Forecast Metadata Conventions*) que en este momento está en la versión 1.6 (cfconventions.org). La versión actual de netCDF es la 4.1 (2010) que permite el uso del formato *HDF5* (hdfgroup.org) y tiene soporte para los lenguajes de programación *C* y *Fortran*. Una ventaja de usar el entorno R es la adaptación a distintas versiones lo que en otros programas puede provocar problemas.

1.2. Fuente

Las fuentes de datos netCDF se encuentran en distintas organizaciones y organismos públicos, así como en grupos científicos de muchos países. En la comunidad Unidata (unidata.ucar.edu/software/netcdf/usage.html) se recoge una copiosa selección de fuentes. Todos aquellos se han adaptado a usar el formato netCDF como estándar para la representación de sus datos espacio-temporales. En climatología destaca el *European Climate Assessment & Dataset Project* (eca.knmi.nl) que permite la descarga de datos climáticos de Europa en cuadrícula en el formato netCDF. Otra fuente es la agencia federal estadounidense *National Oceanic and Atmospheric Administration* (NOAA): esrl.noaa.gov/psd/data/gridded.

En este trabajo usaremos la base de datos de España de las temperaturas máximas creada por el grupo de investigación de Meteorología de la Universidad Cantabria (Herrera et al., 2015). Esta base de datos llamada "Spain02", versión 4, incluye una cuadrícula de diferentes resoluciones (0,11º; 0,22º y 0,44º) para el periodo 1971-2007 de la precipitación diaria y la temperatura mínima y máxima: meteo.unican.es/en/datasets/spain02. Para la elaboración de estas cuadrículas se han usado diferentes metodologías de interpolación (Herrera et al., 2015). En este artículo haremos uso de la cuadrícula de 0,11º de la temperatura máxima, interpolada con *Ordinary Kriging*: meteo.unican.es/thredds/fileServer/Spain_CORDEXgrids/Spain011/tasmax/Spain011_ok_tasmax.nc.

1.3. Entorno R

Instalación de R y bibliotecas

a) Se descarga y se instala *R GUI* (Figura 2) en el equipo informático: **r-project.org.** Se pueden descargar versiones para Linux, Mac OS y Windows. A continuación se deben instalar y seguir las instrucciones correspondientes del programa.

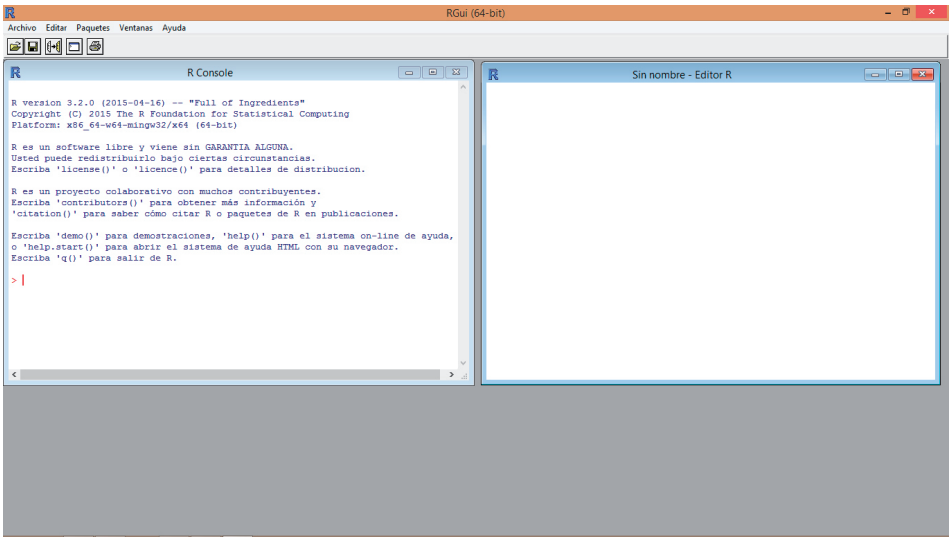


Figura 2. Interfaz de R GUI en Windows.

b) Recomendable, pero opcional, es un interfaz más sofisticado *RStudio* (Figura 3): **rstudio.com**. Se pueden descargar versiones para Linux, Mac OS y Windows.

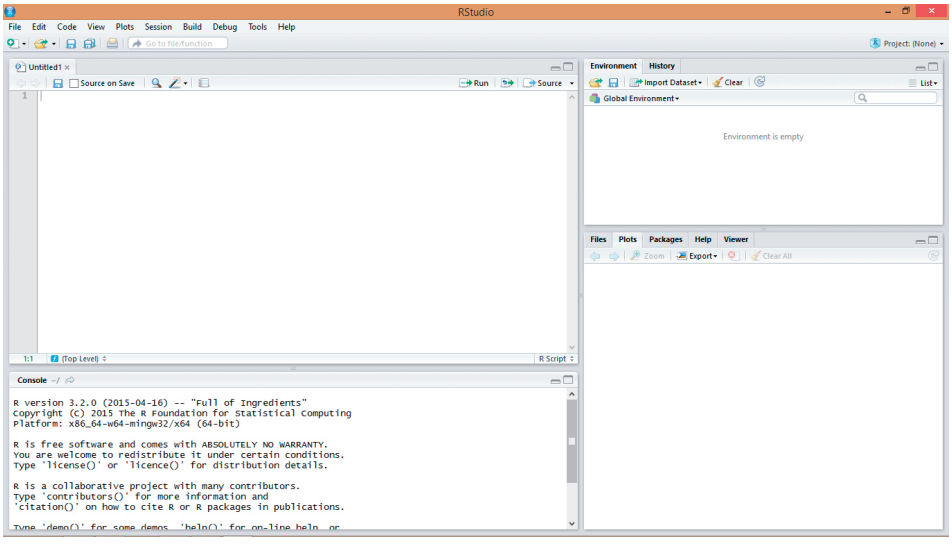


Figura 3. Interfaz RStudio en Windows.

c) Instalar bibliotecas (*packages*):

La única biblioteca adicional necesaria es {*ncdf4*}. No obstante, puede ser útil instalar la biblioteca {*ncdf4.helpers*}. En estos momentos no es posible instalar *ncdf4* directamente en Windows con la forma habitual a través de la función install.packages ("nombre_biblioteca"). Se hace necesario descargar el paquete desde la página web del autor, David Pierce: http://cirrus.ucsd.edu/~pierce/ncdf/.

Para la instalación manual de la biblioteca descargada se abre una sesión en RGui o RStudio. Desde RStudio: Tools/Install Packages/ (Option: from Package Archive File), y desde RGui: Packages/Install Packages from Local File.

Conceptos elementales

El entorno del software libre R se caracteriza por sus múltiples posibilidades. Sobresale su capacidad de almacenamiento y manipulación de datos, pero también los procedimientos estadísticos implementados, la realización de múltiples operaciones matemáticas o la conexión con otros lenguajes de programación y su potente capacidad gráfica. El lenguaje de R está orientado a *objetos* y es de uso común por la comunidad estadística. El uso de R ha experimentado en los últimos años una expansión exponencial más allá de sus orígenes puramente estadísticos (Bivand et al., 2013; Chang, 2013; Crawley, 2012; Paradis, 2003). Una selección más amplia se puede encontrar en http://cran.r-project.org/other-docs.html. En las próximas líneas se abordan algunos aspectos necesarios para el uso de los netCDF en el entorno de R.

En R las órdenes elementales consisten en expresiones o asignaciones. Nótese que el programa nada más imprime el resultado de una operatoria. En todo este trabajo se muestran las funciones (rojo) y sus resultados (negro) con un fondo gris, siempre que existan. El símbolo “>” sindica la línea de comando.

```
> 1+1
[1] 2
```

Una asignación a un objeto, en este caso “x”, se puede realizar con la expresión “<-”. De este modo se guarda el resultado de la operación matemática en “x”. Para conocer el valor guardado se puede usar la función *print( )* o simplemente el objeto mismo “x”.

```
> x <- 1+1
> print(x)
[1] 2
> x
[1] 2
```

Es importante que no se trabaje directamente desde la consola sino desde una ventana de script. Esto hace que todas las funciones no se pierden y pueden ser ejecutadas nuevamente. Una función de una única fila se ejecuta estando con el cursor en la misma línea y el uso de la combinación de tecla *ctrl+R*. En el caso de varias líneas de funciones y comandos es necesario seleccionar todas las líneas.

En R existen varios tipos de objetos: *vector, matrix, data.frame, array* y *list*. Un vector es una cadena de elementos (números, caracteres o valores lógicos). Los elementos de cada objeto están indexados y son seleccionables a través de corchetes.

```
> x <- 10:1
> x
 [1] 10  9  8  7  6  5  4  3  2  1
> x[2]
[1] 9
```

En cambio, una *matrix* tiene dos dimensiones, filas y columnas, y puede contener números, caracteres o valores lógicos. El objeto *data.frame* se caracteriza por la misma estructura, filas y columnas, pero permite incluir en cada columna un tipo distinto. La selección se realiza con la estructura *objeto[i, j]*, siendo *i* la fila y *j* la columna. El índice vacío equivale a utilizar todo el rango de valores para dicha dimensión.

```
> x <- matrix(1:8,nrow=2)
> x
     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
> x[1,]
[1] 1 3 5 7
> x[1,2]
[1] 3
```

Una ampliación de la *matrix* con más de dos dimensiones se encuentra en el objeto de tipo *array*. Por ejemplo, un *array* de tres dimensiones, donde la fila y columna representarían la latitud y longitud, podría incluir el tiempo en su tercera dimensión. Para seleccionar se usa la forma *objeto[i, j, n]*. Este último tipo de objeto será aquel con el que se trabajará en el contexto de netCDF.

```
> x <- array(1:8,c(2,2,2))
> x
, , 1

     [,1] [,2]
[1,]    1    3
```

```
[2,]   2   4

, , 2

     [,1] [,2]
[1,]    5    7
[2,]    6    8
```

En R existe un extenso número de funciones. La estructura es la siguiente: *nombre_función(argumento1, argumento2,…)*. Por ejemplo, la función que calcula el promedio de una secuencia numérica de un vector es *mean( )*. Esta función dispone de un argumento adicional "trim" que permite calcular el promedio truncado.

```
> x <- c(10:1,50)
> x
 [1] 10  9  8  7  6  5  4  3  2  1 50
> mean(x)
[1] 9.545455
> mean(x,trim=0.1)
[1] 6
```

2. PROCESO METODOLÓGICO EN R

2.1. Abrir una conexión

Un paso previo es conocer el directorio de trabajo, para lo que resulta útil la función getwd( ). En este caso el directorio por defecto es "C:/Users/Admin/Documents". Para poder cambiar el directorio se aplica otra función *setwd("nombre_directorio")*. Además es posible usar la función *dir( )* para conocer qué archivos o subcarpetas contiene el directorio y a su vez la carpeta "tamx". Se puede ver que la carpeta contiene la base de datos de España de la temperatura máxima en formato netCDF.

```
> getwd()
[1] "C:/Users/Admin/Documents"
> setwd("C:/Users/Admin/Desktop/tamx")
> getwd()
[1] "C:/Users/Admin/Desktop/tamx"
> dir()
[1] "Spain011_ok_tasmax.nc"
```

A continuación se cargan las bibliotecas.

```
> library(ncdf4)
> library(ncdf4.helpers)
```

Si la biblioteca "ncdf4" o "ncdf4.helpers" están instaladas correctamente no debe haber ningún mensaje de error. Por el contrario, si no están instaladas, *R* devuelve el siguiente aviso:

```
> library(ncdf4)
Error in library(ncdf4) : there is no package called 'ncdf'
```

El ejemplo expuesto sería necesario volver a instalar la biblioteca como se explicó anteriormente.

Con el objetivo de obtener una ayuda global de una biblioteca como la "ncdf4" se puede usar una función que abre una ventana con la descripción básica incluyendo una lista de funciones (Figura 4).

```
> help(package="ncdf4")
```

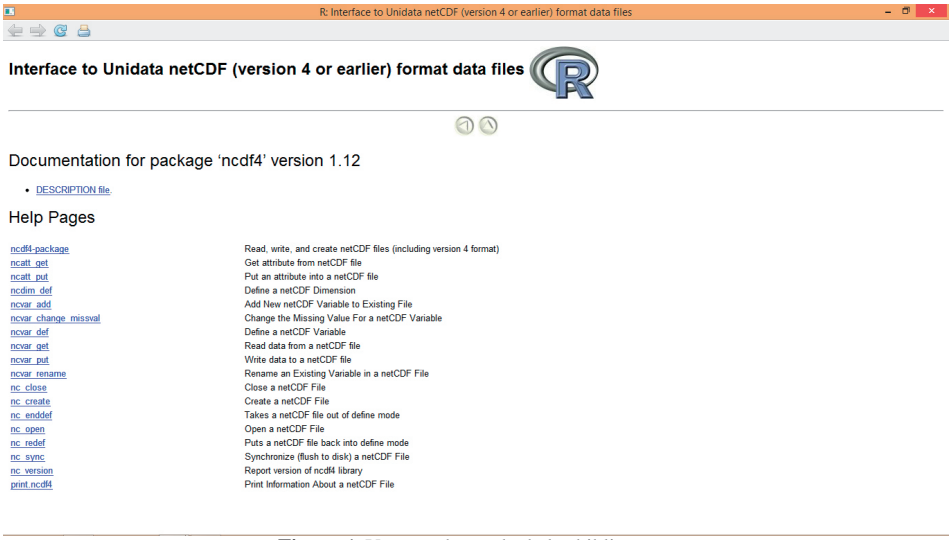


Figura 4. Ventana de ayuda de las bibliotecas.

Además de la ayuda global de una biblioteca existe la posibilidad de conocer los detalles de cada función con la forma *?nombre_función*. Por ejemplo, los argumentos y la descripción concreta sobre *nc_open( )*, una función para abrir la conexión con la base de datos netCDF, se puede obtener de la siguiente forma (Figura 5).

```
> ?nc_open
```

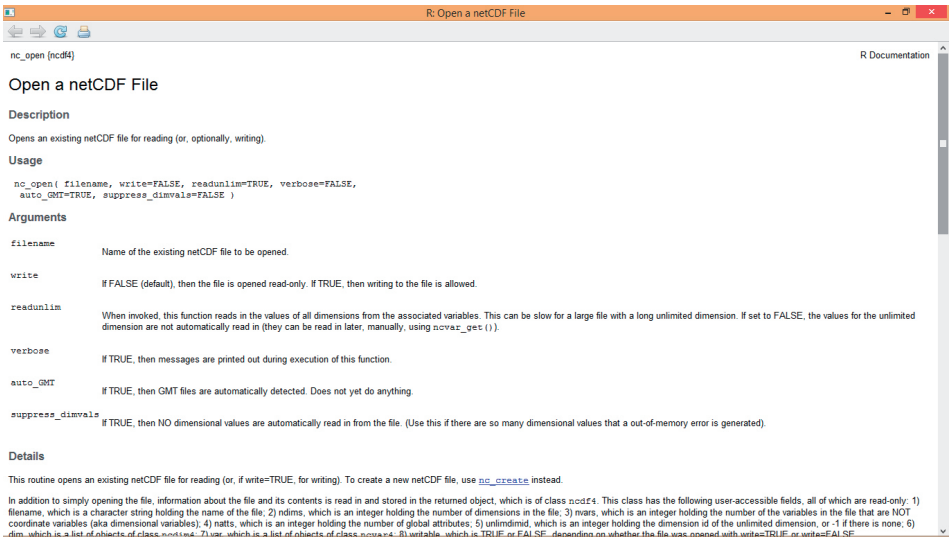


Figura 5. Ventana de ayuda de una función.

Antes de poder trabajar con los datos, es necesario abrir una conexión a la base de datos. Se asigna al objeto "nc" la conexión con los metadatos que necesita R para leer su contenido.

```
> nc <- nc_open("Spain011_ok_tasmax.nc")
```

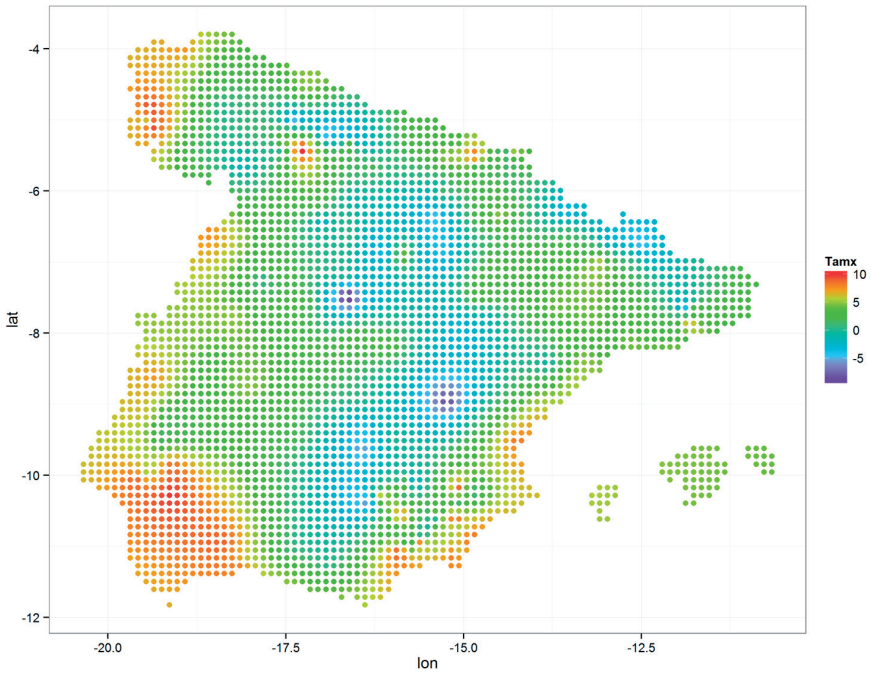
Si se quieren ver los metadatos que contiene "nc", únicamente se debe usar como orden el objeto mismo "nc".

```
> nc
[1] "File Spain011_ok_tasmax.nc (NC_FORMAT_CLASSIC):"
[1] ""
[1] "     3 variables (excluding dimension variables):"
[1] "        double lat[rlon,rlat]   "
[1] "            standard_name: latitude"
[1] "            long_name: latitude"
[1] "            units: degrees north"
[1] "            _CoordinateAxisType: Lat"
[1] "        double lon[rlon,rlat]   "
[1] "            standard_name: longitude"
[1] "            long_name: longitude"
[1] "            units: degrees east"
[1] "            _CoordinateAxisType: Lon"
[1] "        float tasmax[rlon,rlat,time]   "
[1] "            standard_name: air_temperature"
```

```
[1] "        long_name: Daily maximum 2-meters air temperature"
[1] "        units: degrees celsius"
[1] "        missing_value: -9999"
[1] "        coordinates: lat lon time"
.
.
.
.
(acortado)
```

Este resumen de los metadatos sirve para identificar la estructura global de los datos. Se puede ver que *Spain011_ok_tasmax.nc* consiste en 3 variables; latitud, longitud y temperatura máxima de aire: tasmax[rlon, rlat, time]. Además existe una tercera dimensión "time" en la matriz. En este caso representa la fecha entre los años 1971 y 2007. Es importante indicar que esta base de datos en su versión 4 es proyectada en la posición geográfica real (Fig. 6) para coincidir con la cuadrícula del Euro-CORDEX (Herrera et al., 2015). Asimismo se identifican los nombres de las variables en su formato largo y corto, la unidad de cada una de las variables (grados, grados celsius) y la codificación para valores ausentes (-9999).

A)



B)

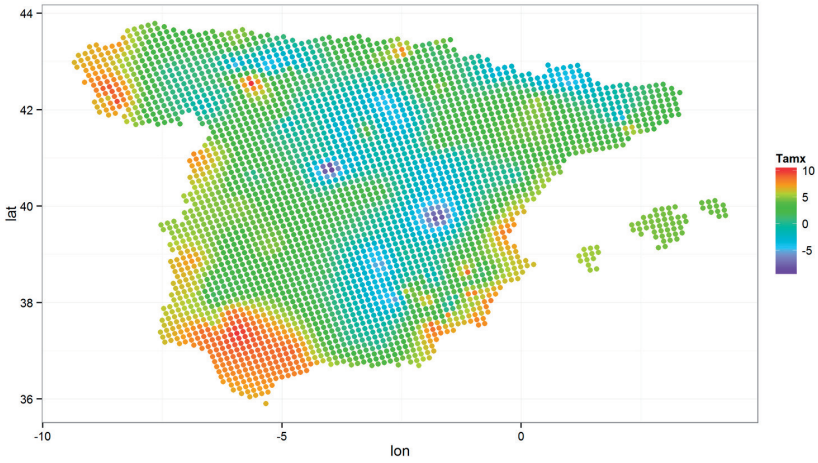


Figura 6. Proyecciones. A) posición geográfica real, B) posición volteada.

2.2. Importar e identificar información

Metadatos

Las siguientes funciones pertenecen a la biblioteca {ncdf4.helpers}. Con ellas se pueden extraer, por un lado, el nombre de la variable principal y la temperatura máxima, y por otro, los nombres de las dimensiones: fecha, latitud y longitud.

```
> nc.get.variable.list(nc)
[1] "tasmax"
> nc.get.dim.names(nc)
[1] "time" "rlat" "rlon"
```

Una función muy útil es *str( )* que permite ver la estructura de un objeto en el entorno R.

```
> str(nc)
List of 14
 $ filename   : chr "Spain011_ok_tasmax.nc"
 $ writable   : logi FALSE
 $ id         : int 196608
 $ safemode   : logi FALSE
 $ format     : chr "NC_FORMAT_CLASSIC"
 $ is_GMT     : logi FALSE
 $ groups     :List of 1
  ..$ :List of 7
  .. ..$ id   : int 196608
  .. ..$ name : chr ""
```

```
 .. ..$ ndims: int 3
 .. ..$ nvars: int 6
 .. ..$ natts: int 7
 .. ..$ dimid: int [1:3(1d)] 0 1 2
 .. ..$ fqgn : chr ""
 .. ..- attr(*, "class")= chr "ncgroup4"
.
.
.
(acortado)
```

Datos matriciales

En el siguiente paso se importan los datos de la temperatura del aire. Para poder cargar toda la matriz de datos diarios (14610 días o 40 años) es necesario tener más de 3 GB de memoria RAM.

```
> data <- ncvar_get(nc,"tasmax")
> str(data)
 num [1:89, 1:85, 1:14610] NA NA NA NA NA NA NA NA NA NA ...
```

Se observa una estructura matricial (*array*) con datos numéricos (*num*) [fila, columna, fecha], correspondiéndose la fila con la latitud y la columna con la longitud. Los valores "NA" (*not available*) visibles en el extracto superior no deben ser confundidos con valores ausentes, sirven para rellenar la matriz y obtener una cuadrícula regular.

Si solo se persigue importar una parte, bien por no tener suficiente memoria o bien por otro motivo, se puede indicar un período concreto.

```
> data2 <- ncvar_get(nc,"tasmax", start=c(1,1,1),count=c(-1, -1, 360))
> str(data2)
 num [1:89, 1:85, 1:360] NA NA NA NA NA NA NA NA NA NA ...
```

La idea se basa en que se indique un punto de inicio *start=c(fila, columna, índice de fecha)* y después cuantos días debe contar para importar *count=c(fila, columna, número de días)*. La indicación *-1* representa a todos los valores de la dimensión correspondiente. En el ejemplo dado, se importan los datos desde el primer día de la serie temporal hasta los 360 días, lo que sería la serie comprendida entre el 01-01-1971 y el 25-12-1971.

Respecto a los valores ausentes es importante conocer los atributos de la variable principal en la "tasmax". Algunos de los atributos se vieron anteriormente. Los valores ausentes son codificados con el valor -9999.

```
> ncatt_get(nc,"tasmax")
$standard_name
[1] “air_temperature”

$long_name
[1] “Daily maximum 2-meters air temperature”

$units
[1] “degrees celsius”

$missing_value
[1] -9999

$coordinates
[1] “lat lon time”
```

En R se pueden realizar fácilmente consultas lógicas como en el siguiente apartado, en el que se condiciona cada celda de la matriz si es igual al valor -9999. R devuelve como resultado otra matriz con valores FALSE o TRUE (valores NA siguen siendo NA).

Valores ausentes

```
> data==-9999
   [,78] [,79] [,80] [,81] [,82] [,83] [,84] [,85] ...
 [1,]    NA    NA    NA    NA    NA    NA    NA    NA
 [2,]    NA    NA    NA    NA    NA    NA    NA    NA
 [3,]    NA    NA    NA    NA    NA    NA    NA    NA
 [4,]    NA    NA    NA    NA    NA    NA    NA    NA
 [5,]    NA    NA    NA    NA    NA    NA    NA    NA
 [6,]    NA    NA    NA    NA    NA    NA    NA    NA
 [7,] FALSE    NA    NA FALSE FALSE    NA    NA    NA
 [8,] FALSE FALSE FALSE FALSE FALSE FALSE    NA    NA
 [9,] FALSE FALSE FALSE FALSE FALSE FALSE    NA    NA
[10,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE    NA
[11,] FALSE FALSE FALSE FALSE FALSE FALSE FALSE    NA
[12,] FALSE FALSE FALSE FALSE FALSE FALSE    NA    NA
[13,] FALSE FALSE FALSE FALSE FALSE FALSE    NA    NA
[14,] FALSE FALSE FALSE FALSE FALSE FALSE    NA    NA
.
.
.
(acortado)
```

Teniendo en cuenta que FALSE corresponde a un valor de 0 y TRUE a 1, la suma total aporta el número de valores ausentes. El argumento en la siguiente función “na.rm=TRUE” es importante porque permite excluir los NA. Finalmente, como se puede ver en este caso, no se encuentran valores ausentes en la base de datos.

```
> sum(data==-9999,na.rm=TRUE)
[1] 0
```

Si hubiera valores ausentes se deben sustituir los -9999 por NA.

```
> data[data==-9999] <- NA
```

Coordenadas

La importación de las coordenadas se realiza siguiendo el mismo procedimiento que se aplicó para la variable principal. Las coordenadas están presentes en el formato de la proyección real y volteado indicado con el nombre "rlon" y "rlat" o "lon" y "lat", respectivamente.

```
> lon <- ncvar_get(nc,"lon")
> lat <- ncvar_get(nc,"lat")
> head(lat)
 [,1]    [,2]    [,3]    [,4]    [,5]...
[1,] 34.37934 34.48331 34.58727 34.69122
[2,] 34.41421 34.51825 34.62226 34.72627
[3,] 34.44892 34.55302 34.65709 34.76115
[4,] 34.48346 34.58762 34.69175 34.79587
[5,] 34.51784 34.62205 34.72624 34.83042
[6,] 34.55205 34.65631 34.76057 34.86480
.
.
.
> rlon <- ncvar_get(nc,"rlon")
> rlat <- ncvar_get(nc,"rlat")
```

A la finalización del uso de la conexión con la base de datos netCDF es posible cerrar esta con la función *nc_close( )*.

2.3. Dimensión temporal: fechas

Las fechas se pueden importar de una forma similar a la variable principal con la función *ncvar_get( )*. Únicamente se cambia el nombre "tasmax" por "time" que incluye la dimensión temporal, concretamente la serie comprendida entre el 1 de enero de 1971 y el 31 de diciembre de 2010. Además se usa la función *ncatt_get( )* para conocer los detalles sobre esta dimensión.

```
> fechas <- ncvar_get(nc,"time")
> str(fechas)
 num [1:14610(1d)] 7670 7671 7672 7673 7674 ...
> ncatt_get(nc,"time")
$long_name
[1] "Time variable"

$units
[1] "days since 1950-01-01 00:00:00"

$`_CoordinateAxisType`
[1] "Time"
```

Puede observarse como el formato de las fechas corresponde al número de días desde el 1 de enero de 1950. Esto requiere una transformación porque el formato de las fechas por defecto en R es "año-mes-día" (por ej. "2000-01-01"). La función *as.Date( )* puede convertir fechas de múltiples formas, por ejemplo, "01/02/2000" en un objeto entendible por R. No obstante, en este caso se tiene un formato numérico que requiere un argumento adicional, *origin* (la fecha a partir del cual se ha contado el número de días).

```
> fechas <- as.Date(fechas,origin="1950-01-01")
> str(fechas)
 Date[1:14610], format: "1971-01-01" "1971-01-02" "1971-01-03" "1971-01-04" "1971-01-05" "1971-01-06" ...
> range(fechas)
[1] "1971-01-01" "2010-12-31"
```

Una vez convertida la fecha en un objeto de clase *Date* se pueden realizar cálculos con ellas. La función *range( )* devuelve el valor mínimo y máximo, en este caso la primera y última fecha de la serie. Otro ejemplo, sería la aplicación de la función *diff( )* que calcula la diferencia en días entre ambas fechas (14609 días).

```
> diff(range(fechas))
[1] 14609
```

Visualización de una única fecha

Primero, se recuerda que la estructura de la matriz es [fila, columna, fecha]. Para obtener la posición, o sea, el índice dentro de la matriz, se puede usar una consulta lógica de la función *which( )*. Además, se puede aplicar una función gráfica *filled.contour( )* que permita visualizar los datos matriciales (Figura 7). Importante es que el primer argumento ha de ser una matriz de dos dimensiones, para ello se introduce la consulta lógica en la parte de la dimensión temporal. El argumento adicional, pero no necesario, *color.palette*, permite definir el rango de colores que usa la función gráfica (más ayuda: *?filled.contour* y *?rainbow*).

```
> which(fechas=="1971-01-01")
[1] 1
> which(fechas=="2006-08-01")
[1] 12997
> filled.contour(data[,,which(fechas=="2006-08-01")],
color.palette=rainbow)
```

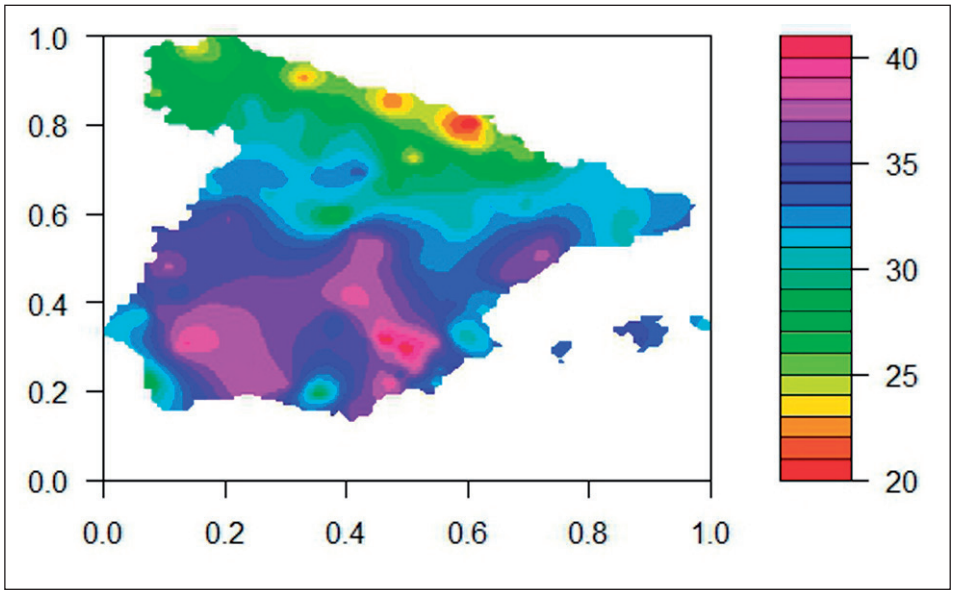


Figura 7. Visualización de las temperaturas máximas del 1 de agosto de 2006.

Visualización de una serie temporal

Segundo, la visualización anterior (Figura 7) ha sido espacial pero también es posible extraer una serie temporal del *array*. Por ejemplo, la serie temporal de la fila 17 y columna 72 se corresponde con las coordenadas -7,42242 y 42,32544. Geográficamente es un punto localizado al sur del río Sil (Ourense). Para obtener un resultado gráfico de la serie temporal (Figura 8) se usa la función *plot( )*, del sistema grafico básico en R. La diferencia entre la primera función indicada y la segunda es la incorporación de la fecha "*data.frame(fechas,data[17,72,])*". "Fechas" es el objeto creado en el apartado 2.3.

```
> plot(data[17,72,],type="l",ylab="°C")
> plot(data.frame(fechas,data[17,72,]),type="l",ylab="°C")
```

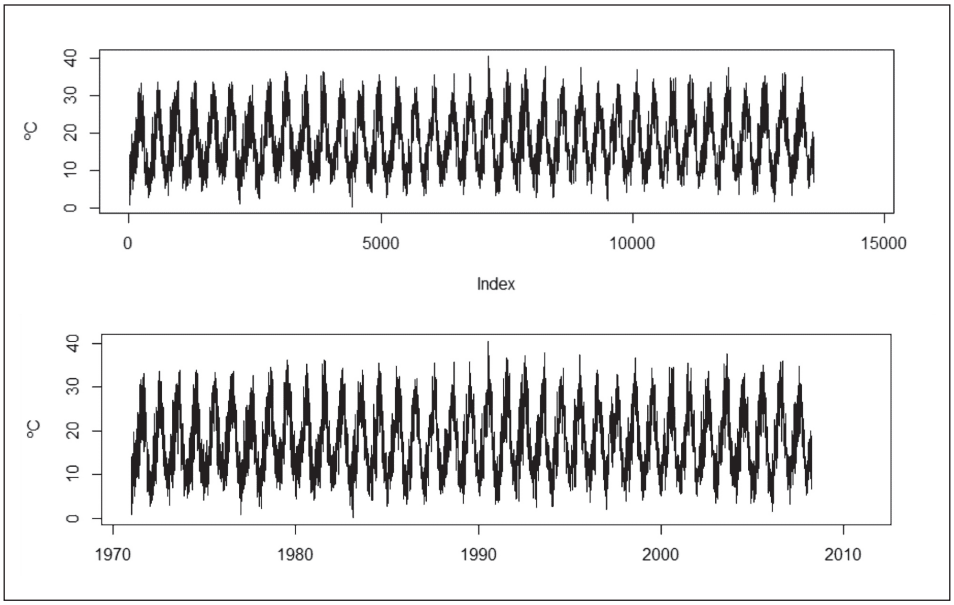


Figura 8. Visualización de una serie temporal de las temperaturas máximas (1971-2007) en un punto con longitud -7,42242 y latitud 42,32544.

2.4. Cálculos globales sobre toda la serie temporal

Con el objetivo de realizar cálculos en toda la serie temporal, se hace uso de una función esencial en R, *apply( )*, que permite aplicar otras funciones a las distintas dimensiones de una matriz. En este primer ejemplo se calcula el promedio de las máximas diarias para la serie temporal, asignando el resultado al objeto "medianual".

```
> medianual <- apply(data,c(1,2),function(x)
ifelse(all(is.na(x)),NA,mean(x,na.rm=TRUE)))
```

Todos los argumentos y las partes de la función aplicada se pueden ver con explicaciones en la Figura 9. Se debe señalar que se aplica la función *mean( )* sobre los márgenes de la matriz, teniendo en cuenta toda la serie temporal de cada combinación de latitud y longitud. Es la razón del segundo argumento *c(1,2)* que indica los márgenes (más ayuda: *?c*). Además es necesario introducir la función *ifelse( )* para evitar la aplicación del cálculo sobre combinaciones de latitud y longitud inexistentes de tipo NA, que únicamente corresponden al relleno para la cuadrícula regular.

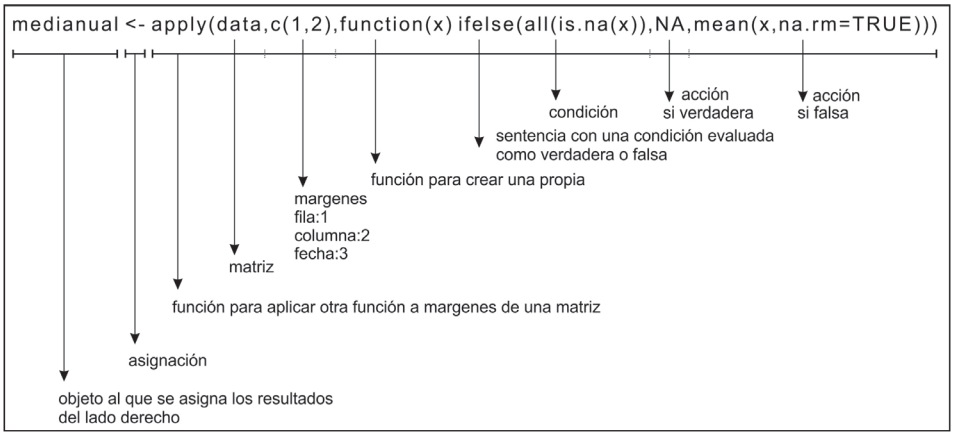


Figura 9. Estructura de la función para cálculos globales con *apply( )*.

Una vez efectuados esos comando se puede introducir el nuevo objeto creado "medianual" en la función gráfica y visualizar el resultado (Figura 10).

```
> filled.contour(medianual,color.palette=rainbow)
```

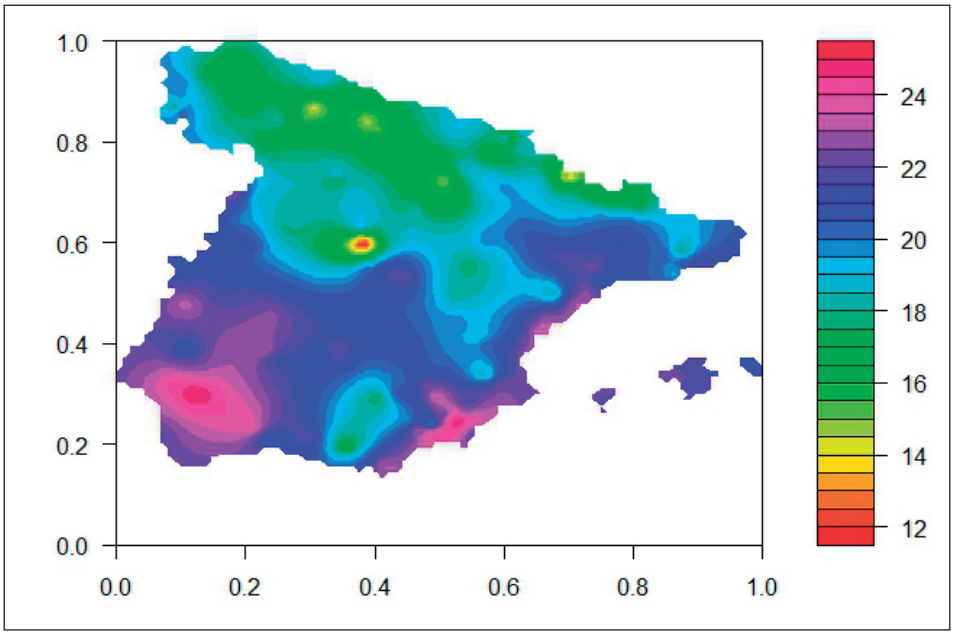


Figura 10. Promedio anual de las máximas diarias.

Otro ejemplo podría ser el número de días con temperaturas superiores a 25°C, conocido como días de verano (ECAD, 2015). Se usa la función *sum( )* con una condición de TRUE-FALSE (x>25) y se divide por el número de años para obtener el promedio anual de días de verano (Figura 11).

```
> diasverano <- apply(data,c(1,2),function(x)
ifelse(all(is.na(x)),NA,sum(x>25,na.rm = TRUE)/40))
> filled.contour(diasverano,color.palette=rainbow)
```

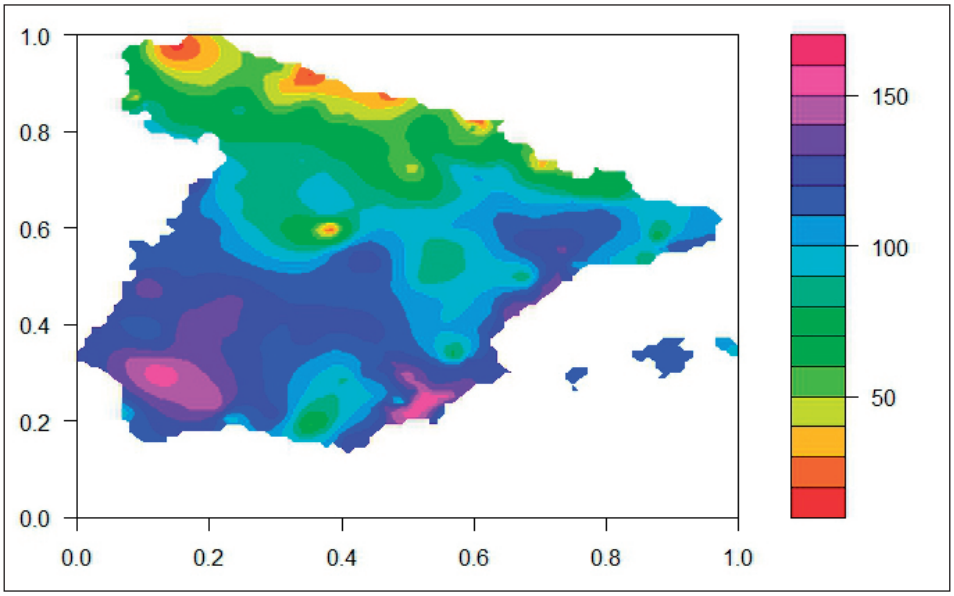


Figura 11. Promedio anual de días de verano.

2.5. Cálculos con agrupaciones temporales

Para el cálculo con agrupaciones temporales primero es preciso generar una variable para la agrupación en partes temporales, por ejemplo por mes. La función *seq( )* genera una secuencia de números o también de fechas. Las indicaciones necesarias van desde "1971-01-01" hasta "2010-12-31", además del intervalo requerido, que en este caso es de 1 día. Para trabajar con otros formatos como "año-mes-día" sería necesario indicar el formato, lo que es posible con el argumento "format", resultando por ejemplo, *as.Date("01/01/1971", format="%d/%m/%Y")*. Más detalles sobre la función *as.Date( )* puede encontrase con *?as.Date* y *?strftime*.

```
> d <- seq(as.Date("1971-01-01"),as.Date("2010-12-31"),1)
> head(d)
```

```
[1] "1971-01-01" "1971-01-02" "1971-01-03" "1971-01-04" "1971-01-05"
[6] "1971-01-06"
```

Siguiente paso consiste en extraer el mes de las fechas, lo que se puede realizar a través de la función *strftime( )*, siendo "d" el objeto de la serie temporal. Las fechas se pueden convertir en caracteres únicamente constituidos por los meses a través del formato "%m" (m: *month*). Se usa la función *as.factor( )* para convertir los caracteres en un objeto de tipo factor. Un factor es una forma utilizada para especificar una clasificación discreta *(?factor*).

```
> strftime(d,format="%m")
  [1] "01" "01" "01" "01" "01"...
> group <- as.factor(strftime(d,format="%m"))
> str(group)
 Factor w/ 12 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Al igual que en los cálculos anteriores se aplica una función a los márgenes, fila y columna *c(1, 2)*, de la matriz "data". Para esta operatoria se hacen necesarias dos funciones adicionales. La primera función es *by( )* y permite usar otra función según agrupaciones, en este caso los meses. La segunda función es *aperm( )* y es empleada para transponer el resultado y obtener el formato de entrada [fila,columna,mes].

Primero se calcula el promedio de las máximas en cada mes del periodo 1971-2010.

```
> tames <- aperm(apply(data,c(1,2), by, group,function(x)
ifelse(all(is.na(x)),NA,mean(x,na.rm=TRUE))),c(2,3,1))
> str(tames)
 num [1:89, 1:85, 1:12] NA NA NA NA NA NA NA NA NA NA ...
 - attr(*, "dimnames")=List of 3
  ..$ : NULL
  ..$ : NULL
  ..$ : chr [1:12] "01" "02" "03" "04" ...
```

El resultado final es una matriz de tres dimensiones con latitud, longitud y los 12 meses. La temperatura media de las máximas en enero y agosto quedaría del modo que sigue (Figura 12 y 13).

```
> filled.contour(tames[,,8],color.palette=rainbow)
> filled.contour(tames[,,1],color.palette=rainbow)
```

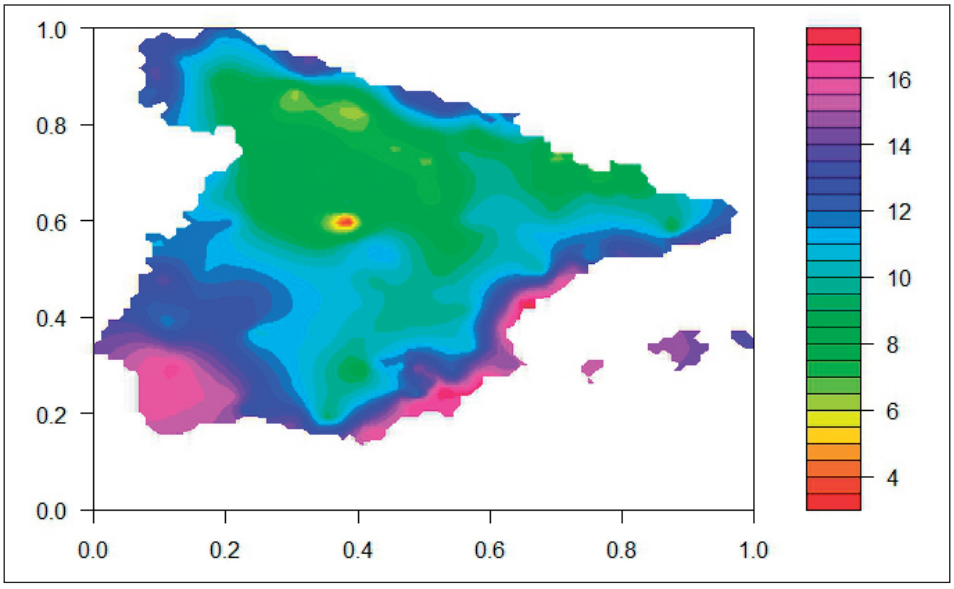


Figura 12. Promedio de las temperaturas máximas diarias en enero (1971-2010).

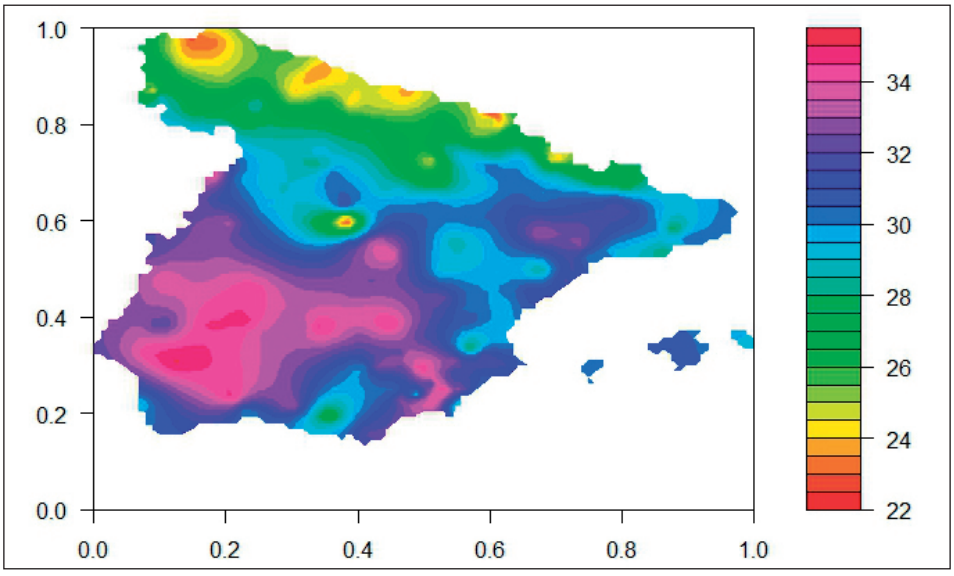


Figura 13. Promedio de las temperaturas máximas diarias en agosto (1971-2010).

Otro ejemplo de una aplicación consiste en el cálculo del valor máximo absoluto de cada mes de la serie temporal (Figura 14). En este caso se sustituye la función *mean( )* por *max( )* y se aplica el mismo método que se hizo en el paso anterior.

```
> tamxabs <- aperm(apply(data,c(1,2), by, group,function(x)
ifelse(all(is.na(x)),NA,max(x,na.rm=TRUE))),c(2,3,1))
> filled.contour(tamxabs[,,1],color.palette=rainbow)
```

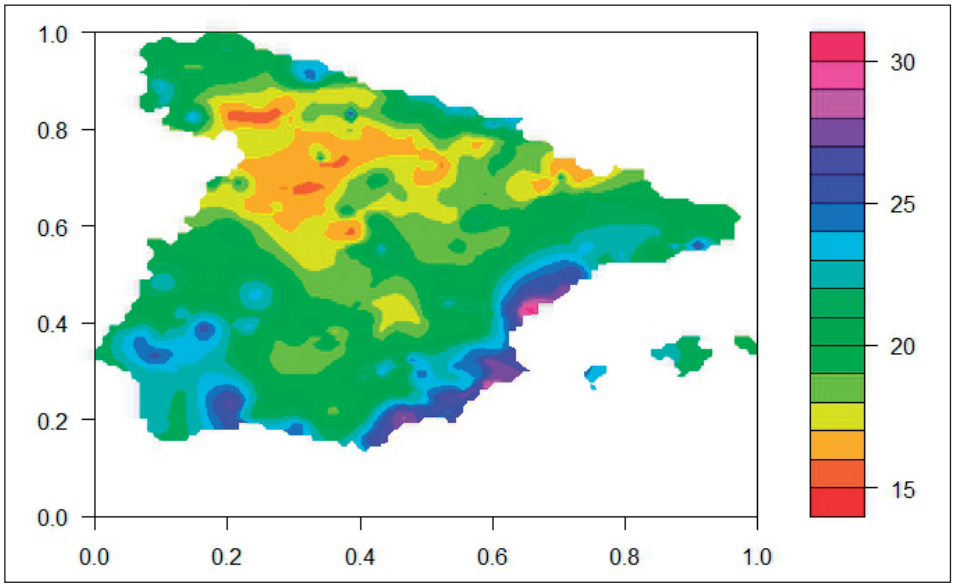


Figura 14. Temperaturas máximas absolutas en enero 1971-2010.

3. CASO APLICADO: LA AMPLITUD DIURNA EN ESPAÑA (ENERO Y AGOSTO 2006)

El objetivo de este apartado es mostrar la aplicabilidad a raíz de un problema concreto de investigación. Se pretende visualizar la media de las amplitudes térmicas diarias en España para los meses de agosto y enero del año 2006. Para ello son necesarias las bases de datos netCDF *Spain011 (OK)* con las temperaturas máximas y mínimas.

En esta práctica se amplían las bibliotecas usadas anteriormente con {lubridate} y {ggplot2}.

```
> library(ncdf4)
> library(lubridate) #para tratar fechas y horas
> library(ggplot2) #salida gráfica
```

En el primer paso se abre una conexión a cada una de las dos bases de datos netCDF.

```
> nc_mx <- nc_open("Spain011_ok_tasmax.nc")
> nc_min <- nc_open("Spain011_ok_tasmin.nc")
```

Seguidamente se importan las fechas para poder seleccionar el periodo de estudio, desde el 1 de enero de 2006 hasta el 31 de diciembre de 2006. De este modo no es necesario importar toda la matriz. Los objetos "st" (start) y "en" (end) sirven para guardar la posición de la fecha de inicio y finalización.

```
> fechas <- ncvar_get(nc_mx,"time")
> fechas <- as.Date(fechas,origin="1950-01-01")
> st <- which(fechas=="2006-01-01")
> en <- which(fechas=="2007-01-01")
```

En el siguiente paso se importan los datos como matrices asignándoles los objetos "data_mx" para las máximas y "data_min" para las mínimas. Es posible usar directamente las posiciones guardadas en los objetos.

```
> data_mx <- ncvar_get(nc_mx,"tasmax",start=c(1,1,st),
count=c(-1, -1,en-st))
> data_min <- ncvar_get(nc_min,"tasmin",start=c(1,1,st),
count=c(-1, -1,en-st))
```

Se importan también las coordenadas por motivos de visualización posterior.

```
> lon <- ncvar_get(nc_mx,"lon")
> lat <- ncvar_get(nc_mx,"lat")
```

Para calcular las amplitudes se debe sustraer la matriz de las mínimas de la matriz de las máximas. La matriz resultante de esta sustracción se asigna al objeto "data_at".

```
> data_at <- data_mx-data_min
```

El siguiente paso se crea el objeto "group" para poder calcular las medias de las amplitudes diarias de cada mes del año 2006. Para ello se hace uso de los objetos "st" y "en" para seleccionar el período. El comando usado ":" [st:en] produce una secuencia de números enteros.

```
> group <- month(fechas)[st:en][-366]
> mean_at <- aperm(apply(data_at,c(1,2), by, group,function(x)
ifelse(all(is.na(x)),NA,mean(x,na.rm=TRUE))),c(2,3,1))
```

El cálculo de las amplitudes térmicas conduce a la visualización de los resultados. Se procede con la asignación de la matriz de agosto y enero a un nuevo objeto "m8" y "m1", respectivamente. Dado que la biblioteca gráfica {ggplot2} requiere una estructura de tabla con las columnas latitud, longitud y la variable, es necesario transformar las matrices.

```
> m8 <- mean_at[,,8] #agosto
> m8 <- data.frame(as.vector(lon),as.vector(lat),as.vector(m8))
> names(m8) <- c("lon","lat","at")
> m8 <- m8[complete.cases(m8),]
>
> m1 <- mean_at[,,1] #enero
> m1 <- data.frame(as.vector(lon),as.vector(lat),as.vector(m1))
> names(m1) <- c("lon","lat","at")
> m1 <- m1[complete.cases(m1),]
```

Antes de continuar se crea una tabla con las localizaciones de cinco ciudades con el objetivo de facilitar la orientación en el mapa resultante.

```
> tab <- data.frame(la=c(40.4125,41.4,43.466667,37.178056,42.33),
+ lon=c(-3.703889,2.166667,-3.8,-3.6,7.87),
+ ciudad=c("Madrid","Barcelona","Santander","Granada","Ourense"))
```

Para la visualización gráfica se usa la biblioteca {ggplot2} que se caracteriza por una estructura de objetos combinados por "+". Se inicia con la definición de la tabla y las variables para los ejes en *ggplot(m8,aes(lon,lat))* y se complementa con la geometría deseada, en este caso, *geom_point(aes(color=at),size=4.8,shape=15)*. Más detalles sobre cómo construir gráficos se pueden encontrar en http://www.cookbook-r.com/Graphs/ o en Chang (2013). En los dos siguientes bloques se muestra la función para el mes de agosto (Figura 15) y el mes de enero (Figura 16).

```
> ggplot(m8,aes(lon,lat))+geom_point(aes(color=at),size=4.8,shape=15)+
+ geom_point(data=tab,aes(lon,lat),size=3,shape=1)+
+ geom_text(data=tab,size=4,fontface="bold",
+ aes(label=ciudad,x=lon,y=lat-0.3))+
+ coord_equal()+scale_color_gradientn("°C",
  colours=rainbow(20),breaks=seq(4,21.81,2),limits=c(4,22))+
+ scale_y_continuous("Latitud",breaks=seq(35,44,1))+
+ scale_x_continuous("Longitud",breaks=seq(-10,5,1))+
+ theme_bw()+coord_equal()

> ggplot(m1,aes(lon,lat))+geom_point(aes(color=at),size=4.8,shape=15)+
+   geom_point(data=tab,aes(lon,lat),size=3,shape=1)+
+   geom_text(data=tab,size=4,fontface="bold",aes(label=ciudad,x=lon,
```

```
y=lat-0.3))+
+   scale_color_gradientn("°C",colours=rainbow(20),breaks=seq(4,21.81,
2),limits=c(4,22))+
+   scale_y_continuous("Latitud",breaks=seq(35,44,1))+
+   scale_x_continuous("Longitud",breaks=seq(-10,5,1))+
+   theme_bw()+coord_equal()
```

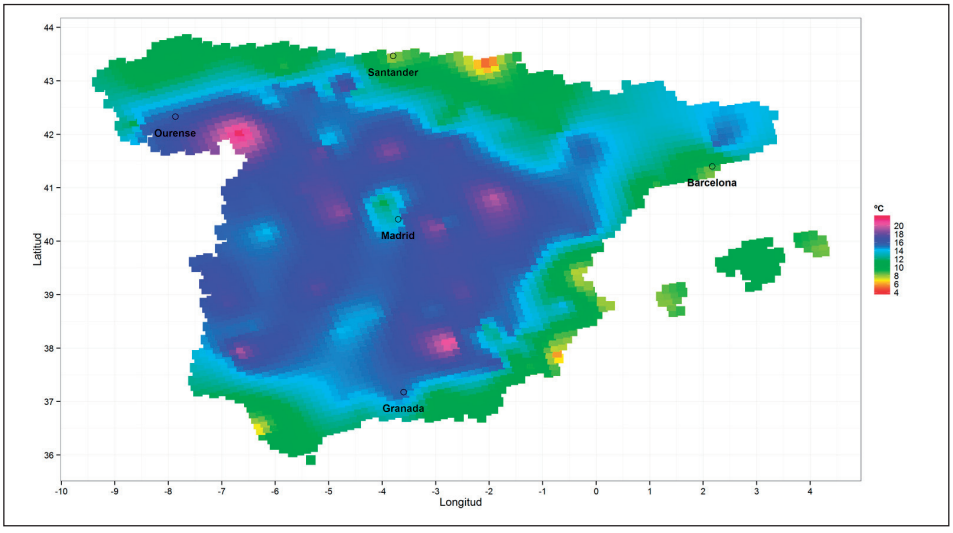


Figura 15. Promedio de la amplitud térmica en agosto de 2006.

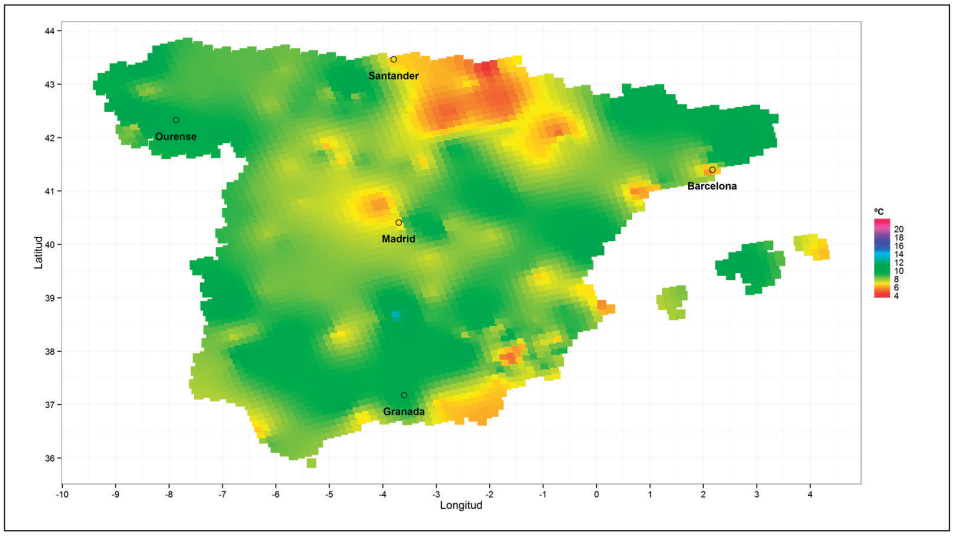


Figura 16. Promedio de la amplitud térmica en agosto de 2006.

Los resultados muestran las amplitudes diurnas para la Península Ibérica en el año 2006 para los meses de agosto y enero. La limitación del año 2006 impide considerar los patrones en las Figuras 15-16 en un sentido climático más amplio.

En agosto las mayores amplitudes se concentran en el interior de la Península debido a su carácter continental. En las zonas costeras se manifiesta el efecto regulador del océano. Se observa una estrecha relación entre la radiación y la amplitud. Las mínimas diurnas están relacionadas con los flujos radiantes de onda larga y las máximas con las de onda corta (Makowski et al., 2008). Otro factor importante es el número de días con cielos despejados, alcanzando las mayores amplitudes en condiciones atmosféricas de estabilidad. Son destacables las amplitudes diurnas máximas en agosto (22ºC) y en enero (13ºC). Destaca también la reducida amplitud térmica en la región de Madrid (Figura 15), provocada presumiblemente por el efecto isla de calor, estrechamente relacionado con la dimensión de la urbe. Como señala López et al. (1993), las ciudades se caracterizan por temperaturas superiores en el núcleo urbano, combinando un lento enfriamiento nocturno en contraste con el medio rural circundante.

En enero el efecto de la nubosidad parece ser más visible (Figura 16). Por ejemplo, destaca la región del País Vasco con una baja amplitud donde el promedio de días despejados se encuentra en coherencia con sólo de 2,6 días (AEMET 2012). La amplitud térmica del mes enero parece evidenciar todavía más los efectos de la isa de calor en algunas ciudades grandes como por ejemplo en Barcelona donde se puede observar valores más bajos en contraste de su entorno. Sería necesario realizar un estudio más detallado y amplio para llegar a conclusiones sobre la distribución espacio-temporal de las amplitudes térmicas en la Península Ibérica.

4. CONSIDERACIONES FINALES

La presente introducción al manejo de netCDF en R muestra de forma breve algunas de las muchas posibilidades que ofrece este entorno de software libre. El formato netCDF es en la actualidad indispensable para poder trabajar con datos a grandes escalas temporales y espaciales.

REFERENCIAS BIBLIOGRÁFICAS

AEMET (2012): *Guía resumida del clima en España (1981-2010)*, http://www.aemet.es/es/conocermas/publicaciones/detalles/guia_resumida_2010.

Bivand, R. S.; Pebesma, E. y Gómez-Rubio, V. (2013[2]): *Applied Spatial Data Analysis with R*, Use R! Serie, Springer.

Chang, W. (2013): *R Graphics Cookbook: Practical Recipes for Visualizing Data*, O'Reilly.

Crawley, M. J. (2012[2]): *The R Book*, Wiley.

ECAD (2015): *E-OBS climate indices for EUPORIAS*. http://www.ecad.eu/download/ensembles/download_R.php

Herrera, S.; Fernández, J. y Gutiérrez, J. M. (2015): "Update of the Spain02 gridded observational dataset for EURO-CORDEX evaluation: assessing the effect of the interpolation methodology", *International Journal of Climatology*, en prensa.

López, A.; Fernández, F.; Arroy, F.; Martin Vide, J. y Cuadrat, J. M. (1993): *El clima de las ciudades españolas*, Catedra.

Makowski, K.; Wild, M. y Ohmura, A. (2008): "Diurnal temperature range over Europe between 1950 and 2005", *Atmos. Chem. Phys.*, 8:6483-6498.

Paradis, E. (2003): *R para Principiantes*, traducido por Jorge A. Ahumada, URL: http://cran.r-project.org/doc/contrib/rdebuts_es.pdf.