Evaluación de los efectos de los escenarios hidrológicos proyectados para el siglo XXI sobre la generación hidroeléctrica

Acción 4

Desarrollo de la herramienta informática para la evaluación de los efectos del cambio climático sobre la generación hidroeléctrica.

R4.1 - Herramienta informática para la evaluación de los efectos del cambio climático sobre la generación hidroeléctrica

Elaborado por Tecnalia

tecnalia MEMBER OF BASQUE RESEARCH & TECHNOLOGY ALLIANCE

Proyecto desarrollado con el apoyo de la Fundación Biodiversidad, del Ministerio para la Transición Ecológica y el Reto Demográfico

Metodología para el análisis de proyecciones de caudales

Introducción al entorno

Bienvenido a la exposición de la metodología para adquirir y tratar proyecciones de caudal aplicables en el análisis de la evolución de la generación hidroeléctrica.

Este página web es un libro de Jupyter que se ejecuta en Google Collab. Google collab es una máquina virtual que facilita su ejecución. Gracias a esta plataforma no tendrás que construir tu propio sistema en tu ordenador personal u otro sistema. Con abrir esta página en un navegador (quizás Google Chrome es el más adecuado) es sufciente.

Los libros de Jupyter son básicamente una combinación de un lenguaje de programación (en este caso Python) y textos, fotos, vídeos, etc. Existend dos tipos de celdas. Estas que está leyendo está diseñada para almacenar texto, imágenes estáticas, vídeos, etc. No se pueden ejecutar. Las que incluyen código, como la incluida a continuación tienen una flecha a la izquierda. Si la pulsas se ejecutarán y desarrollarán diferentes acciones. En cada una de ellas indicamos que procedimiento lanzan con líneas precedidas de "#", que no se ejecutan.

Recomendamos no prestar mucha atención a los detalles del código, ya que para entenderlo totalmente es preciso disponer de unos ciertos conocimientos del lenguaje de programación Python. Lo más interesante es prestar atención a la secuencia que se propone y a la estructura de los datos con los que estamos trabajando.

```
#Esta es una celda de ejemplo de código.
#Pulsa en la flecha de la izquierda y se ejecutará. El resultado debe ser que bajo la misma se imprime un texto.
#Si cambias el texto entre comillas, el resultado cambiará.
print ("Bienvenido a la exposición de la metodología.")
```

```
Bienvenido a la exposición de la metodología.
```

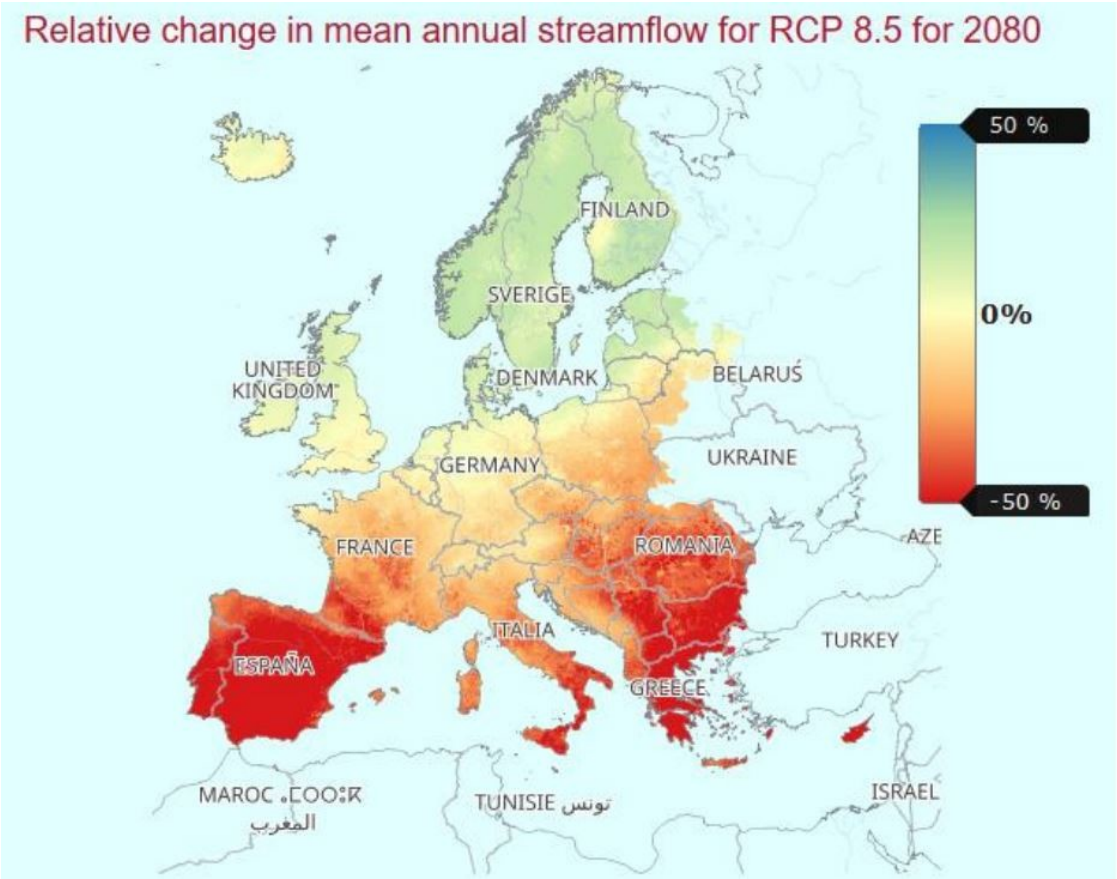
Proyecciones y escenarios climáticos

Pare analizar la previsible evolución de la generación hidroeléctrica, es preciso conocer la evalucion de los caudales. En este ejercicio vamos a emplear los datos aportado por el Servicio de Cambio Climático de Copernicus (C3S). Más concretamente, el conjunto de datos denominado:

"Water sector indicators of hydrological change across Europe from 2011 to 2095 derived from climate simulations."

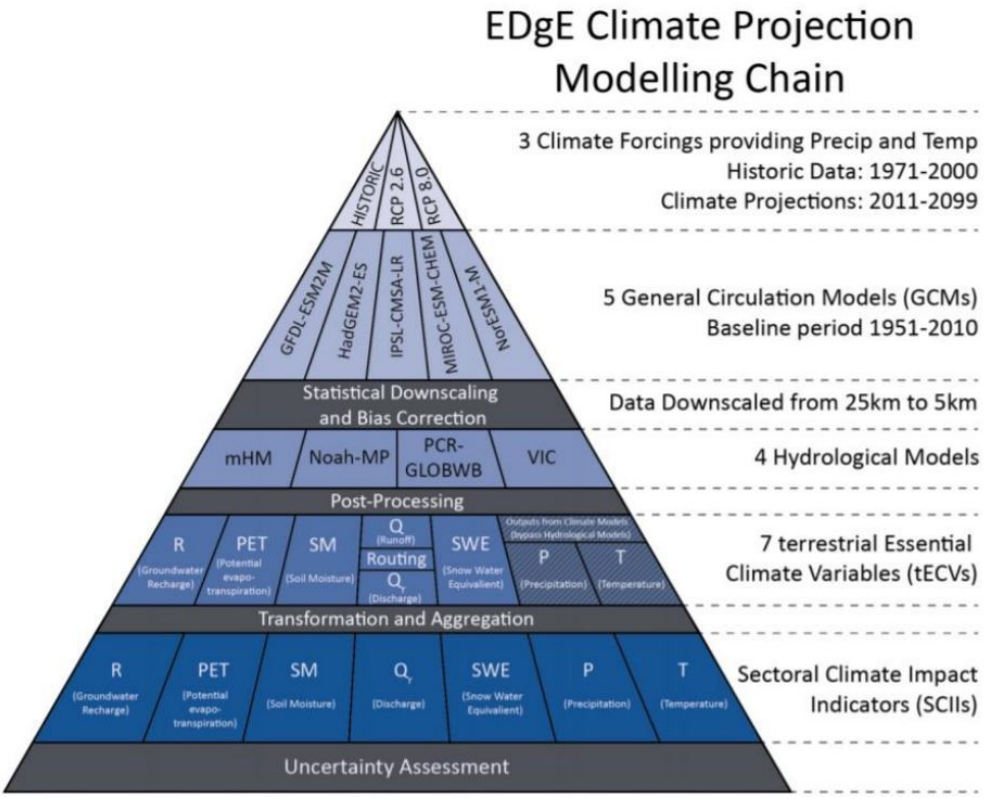
Se encuentra disponible en este enlace.

En la imagen siguiente se aprecia el conjunto de proyecciones que genera para un escenario climático pesimista (que se conoce como RCP8.5)



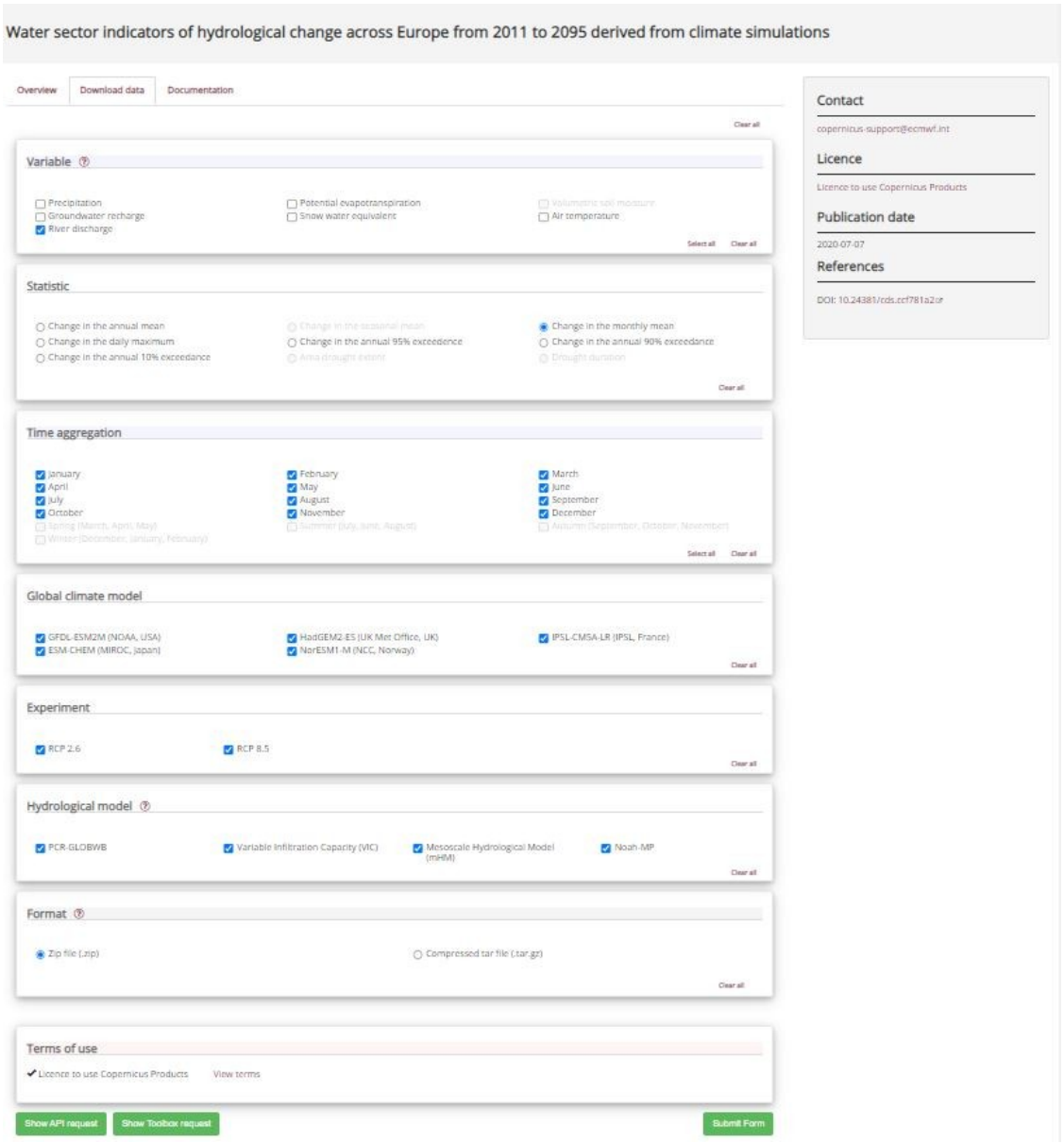
En este libro vamos a exponer como acceder a estos datos y combinarlos con registros locales. Pero antes, vamos a recordar un punto importante.

La generación de proyecciones acerca de la previsible evolución de los impactos del cambio climático se aborda generalmente a través de un conjunto de hipotésis, escenarios y modelos acoplados. Cuando accedemos a estos datos es conveniente tener siempre preseente que realmente estamos utilizando resultados de diferentes combinaciones. En la imagen siguiente se sintetizan las mismas:



Fuente: C3S

De este modo, cuando accedamos a los datos tenemos que tener presente que se han generado para diferentes escenarios económicos, con diferentes modelos climáticos, etc. Esto genera un rango de resultados que nos permite valorar las incertidumbres existentes acerca de la generación de proyecciones climáticas y sus impactos asociados. Es por tanto conveniente descargar resultados para varias de estas combinaciones. En la imagen siguiente te indicamos las que hemos descargado nosotros para generar este ejercicio.



Fuente: C3S

Te recomendamos descargar los mismos datos a tu ordenador personal para poder seguir el ejercicio.

Desafortunadamente la plataforma que vamos a emplear no nos permite gestionar todos los ficheros que es posible descargar desde C3S y vamos emplear solo una parte de ellos: los datos asociados a un único modelo global de circulación: GFDL-ES2M. Para el mismo hemos descargado los datos que se han genenerado con varios escenarios forzamiento radiativo (que son equivalentes a escenarios socioeconómicos), diferentes modelos hidrológicos, etc.

El primer paso para poder acceder a esta información será seleccionar el icono de la carpeta en la parte derecha de tu pantalla y arrastrar todos los archivos que empiecen por

"GFDL-ES2M...". intenta hacerlo de varias veces, porque puede que si no lo haces así el sistema colapse.

#**Importación de las librerías**

Python requiere de diferentes librerías para realizar algunas de las funciones necesarias. En Google Colab tenemos muchas de ellas instaladas. En cambio, en ocasiones tenemos que instalarlas primero. En las próximas celdas realizamos este proceso, que es un poco complicado para alguna librería, ya que hay que instalar, desinstalar, etc.

```
#Esta librería es necesaria para abrir los archivos .nc que hemos
cargado a la izquierda.
!pip install netcdf4
```

```
Requirement already satisfied: netcdf4 in
/usr/local/lib/python3.7/dist-packages (1.5.7)
Requirement already satisfied: cftime in
/usr/local/lib/python3.7/dist-packages (from netcdf4) (1.5.0)
Requirement already satisfied: numpy>=1.9 in
/usr/local/lib/python3.7/dist-packages (from netcdf4) (1.19.5)
```

```
#Ahora vamos a instalar algunas librerías para gestionar información
geográfica. En el próximo paso tendremos que desinstalar una parte de
ellas.
!apt-get install -qq libgdal-dev libproj-dev
!pip install GEOS NumPy Cython Shapely pyshp six
git+https://github.com/SciTools/cartopy.git
#thanks to
#https://github.com/SciTools/cartopy/issues/1346
#https://scitools.org.uk/cartopy/docs/latest/installing.html
```

```
Collecting git+https://github.com/SciTools/cartopy.git
  Cloning https://github.com/SciTools/cartopy.git to /tmp/pip-req-
build-_38e9p72
  Running command git clone -q https://github.com/SciTools/cartopy.git
/tmp/pip-req-build-_38e9p72
  Installing build dependencies ... ents to build wheel ...
etadata ... ent already satisfied: GEOS in
/usr/local/lib/python3.7/dist-packages (0.2.3)
Requirement already satisfied: NumPy in /usr/local/lib/python3.7/dist-
packages (1.19.5)
Requirement already satisfied: Cython in
/usr/local/lib/python3.7/dist-packages (0.29.23)
Requirement already satisfied: Shapely in
/usr/local/lib/python3.7/dist-packages (1.7.1)
Requirement already satisfied: pyshp in /usr/local/lib/python3.7/dist-
packages (2.1.3)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-
packages (1.15.0)
Requirement already satisfied: flask in /usr/local/lib/python3.7/dist-
packages (from GEOS) (1.1.4)
```

```
Requirement already satisfied: lxml in /usr/local/lib/python3.7/dist-
packages (from GEOS) (4.2.6)
Requirement already satisfied: pillow in
/usr/local/lib/python3.7/dist-packages (from GEOS) (7.1.2)
Requirement already satisfied: Werkzeug<2.0,>=0.15 in
/usr/local/lib/python3.7/dist-packages (from flask->GEOS) (1.0.1)
Requirement already satisfied: Jinja2<3.0,>=2.10.1 in
/usr/local/lib/python3.7/dist-packages (from flask->GEOS) (2.11.3)
Requirement already satisfied: click<8.0,>=5.1 in
/usr/local/lib/python3.7/dist-packages (from flask->GEOS) (7.1.2)
Requirement already satisfied: itsdangerous<2.0,>=0.24 in
/usr/local/lib/python3.7/dist-packages (from flask->GEOS) (1.1.0)
Requirement already satisfied: MarkupSafe>=0.23 in
/usr/local/lib/python3.7/dist-packages (from Jinja2<3.0,>=2.10.1-
>flask->GEOS) (2.0.1)
```

```
#Ahora reinstalamos una de las librerías de una manera ligeramente
diferente.
!pip uninstall shapely
!pip install shapely --no-binary shapely
#thanks to https://stackoverflow.com/questions/60111684/geometry-must-
be-a-point-or-linestring-error-using-cartopy
```

```
Found existing installation: Shapely 1.7.1
Uninstalling Shapely-1.7.1:
  Would remove:
    /usr/local/lib/python3.7/dist-packages/Shapely-1.7.1-py3.7.egg-
info
    /usr/local/lib/python3.7/dist-packages/shapely/*
Proceed (y/n)? ERROR: Operation cancelled by user
^C
Requirement already satisfied: shapely in
/usr/local/lib/python3.7/dist-packages (1.7.1)
```

```
#Ahora importaremos las librerías que vamos a usar.
#Si alguna no puede ser importada, sería necesario realizar más "!pip
install XXXX"
import os
import xarray as xr
import matplotlib.pyplot as plt
import cartopy.crs as ccrs
import cartopy.feature as cfeature
from cartopy.mpl.ticker import LongitudeFormatter, LatitudeFormatter
import numpy as np
import pandas as pd
```

Si queremos ver los gráficos en este entorno tenemos que ejecutar la siguiente línea (posiblemente inncecesaria en otros entornos).

```
%matplotlib inline
```

Las siguientes listas indican las combinaciones de modelos y escenarios que podemos manejar. Las hemos acortado por la capacidad de esta plataforma.

```
rcps=['rcp2p6']#si pudieramos cargar todos los RCP la lista sería ['rcp2p6','rcp8p5']
gcm_models=['GFDL-ESM2M'] #si pudieramos cargar todos los ficheros la lista sería ['GFDL-ESM2M','HadGEM2-ES','IPSL-CM5A-LR','MIROC-ESM-CHEM','NorESM1-M']
hidrological_models=['mHM','noah-mp','PCRGLOBWB','VIC']
months=['JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OCT','NOV','DEC']
```

Análisis de uno de los ficheros

Ahora que ya tenemos nuestro entorno configurado, vamos a analizar uno de los ficheros.

```
# Definimos el nombre de nuestro fichero y luego lo abrimos
dataDIR = 'GFDL-ESM2M_rcp2p6_mHM_streamflow_mean-monthly_2011-2095_APR_v1.nc' #add the path if needed.
DS = xr.open_dataset(dataDIR)

# Analizamos los metadatos del archivo, que nos indican que información contiene
print (DS)
print (DS.attrs)
print (DS.dims)
print (DS.coords)
print (DS.time)
```

```
<xarray.Dataset>
Dimensions:                          (month: 1, time: 12, x: 1000, y: 950)
Coordinates:
  * month                            (month) float64 4.0
  * time                             (time) datetime64[ns] 2025-01-01 ... 2080-0...
  * x                                (x) float64 2.502e+06 2.508e+06 ... 7.498e+06
  * y                                (y) float64 5.498e+06 5.492e+06 ... 7.525e+05
Data variables:
    lambert_azimuthal_equal_area  float64 ...
    lat                            (y, x) float64 ...
    lon                            (y, x) float64 ...
    ref_var_threshold              (month, y, x) float32 ...
    relative_change                (time, month, y, x) float32 ...
Attributes:
    title:        SCII-5: Relative change in mean monthly flow over the EDgE ...
```

```
    institution:  Helmholtz Centre for Environmental Research - UFZ
    source:       Routed runoff generated from the mHM model forced
with the ...
    comment:      Relative change in mean monthly flow: The daily flow
is ave...
    Conventions:  CF-1.6
    project:      EDgE - WP2
    processed:    Rohini Kumar (UFZ)
    contact:      rohini.kumar@ufz.de, stephan.thober@ufz.de,
luis.samaniego@...
    history:      Sat Aug  5 06:54:23 2017: C:\nco\ncks -d month,3 D:\
smw\edg...
    NCO:          4.6.7-alpha04
{'title': 'SCII-5: Relative change in mean monthly flow over the EDgE
domain (see comment for further detail)', 'institution': 'Helmholtz
Centre for Environmental Research - UFZ', 'source': 'Routed runoff
generated from the mHM model forced with the GFDL-ESM2M model under
the projection rcp2p6', 'comment': 'Relative change in mean monthly
flow: The daily flow is averaged for each calendar month
(J,F,M,A,...,N,D) over the 30 years of time window. Relative changes
are estimated with respect to the reference value of the period 1971-
2000. Relative changes are calculated over 30 year timeslices with 5
year increments. The metrics is associated with the middle year of the
30 year window. The first period for the relative changes corresponds
to the period 2011-2040', 'Conventions': 'CF-1.6', 'project': 'EDgE -
WP2', 'processed': 'Rohini Kumar (UFZ)', 'contact':
'rohini.kumar@ufz.de, stephan.thober@ufz.de, luis.samaniego@ufz.de',
'history': 'Sat Aug  5 06:54:23 2017: C:\\nco\\ncks -d month,3 D:\\
smw\\edge\\climatechange03\\streamflow-mean-monthly_2011_2095-
unpacked\\GFDL-ESM2M_rcp2p6_mHM_streamflow-mean-monthly_2011_2095.nc
D:\\smw\\edge\\climatechange03\\streamflow-mean-monthly_2011_2095-
unpacked\\GFDL-ESM2M_rcp2p6_mHM_streamflow-mean-monthly_2011_2095-
APR.nc\nFri May 26 2017: created 3rd version', 'NCO': '4.6.7-alpha04'}
Frozen(SortedKeysDict({'y': 950, 'x': 1000, 'month': 1, 'time': 12}))
Coordinates:
  * month    (month) float64 4.0
  * time     (time) datetime64[ns] 2025-01-01 2030-01-01 ... 2080-01-
01
  * x        (x) float64 2.502e+06 2.508e+06 2.512e+06 ... 7.492e+06
7.498e+06
  * y        (y) float64 5.498e+06 5.492e+06 5.488e+06 ... 7.575e+05
7.525e+05
<xarray.DataArray 'time' (time: 12)>
array(['2025-01-01T00:00:00.000000000', '2030-01-
01T00:00:00.000000000',
       '2035-01-01T00:00:00.000000000', '2040-01-
01T00:00:00.000000000',
       '2045-01-01T00:00:00.000000000', '2050-01-
01T00:00:00.000000000',
       '2055-01-01T00:00:00.000000000', '2060-01-
```

```
01T00:00:00.000000000',
       '2065-01-01T00:00:00.000000000', '2070-01-
01T00:00:00.000000000',
       '2075-01-01T00:00:00.000000000', '2080-01-
01T00:00:00.000000000'],
      dtype='datetime64[ns]')
Coordinates:
  * time     (time) datetime64[ns] 2025-01-01 2030-01-01 ... 2080-01-
01
Attributes:
    standard_name:  time
    axis:           T
```

Vamos a extraer los datos que nos interesan (los cambios relativos, es decir, el valor generado para el periodo futuro dividido entre el valor generado para el periodo histórico considerando los mismos escenarios y modelos).

```
#los datos que nos interesan se adquieren asi
relative_changes=DS.relative_change

#ahora veremos que forma tienen los datos descargados
print(relative_changes.shape)

#hay que adquirir también los datos de latitud y longitud
lats=DS.lat.values
print("formato de los datos de latitud:", lats.shape)
lons=DS.lon.values
print("formato de los datos de longitud:", lons.shape)
times=DS.time.values
```

```
(12, 1, 950, 1000)
formato de los datos de latitud: (950, 1000)
formato de los datos de longitud: (950, 1000)
```

Analisis de varios ficheros

Bien, ya que sabemos a que datos nos enfrentamos vamos a extaer datos para varios meses e intentar generar una media anual de los cambios relativos.

```
big_array=np.zeros((12, 12, 950, 1000))
for i in range(12):
    print(months[i])
    dataDIR = 'GFDL-ESM2M_rcp2p6_mHM_streamflow_mean-monthly_2011-
2095_'+months[i]+'_v1.nc' #we are already in the right folder. add the
path if needed.
    DS = xr.open_dataset(dataDIR)
    relative_changes=DS.relative_change.values

    relative_changes.shape
```

```
    relative_changes=np.squeeze(relative_changes)
    relative_changes.shape
    big_array[i]=relative_changes

big_array_mean=np.mean(big_array,axis=0)
print(big_array_mean.shape)
```

```
JAN
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC
(12, 950, 1000)
```

Como vemos, algunos valores son demasiado elevados. Vamos a eliminar los que sean superiores a 100 (que indican que el caudal se dobla en el futuro en relación al presente).

```
data=big_array_mean[11]
print(np.nanmax(data))
print(np.nanmin(data))

big_array_mean[big_array_mean>100]=100
print(np.nanmax(data))
print(np.nanmin(data))
```

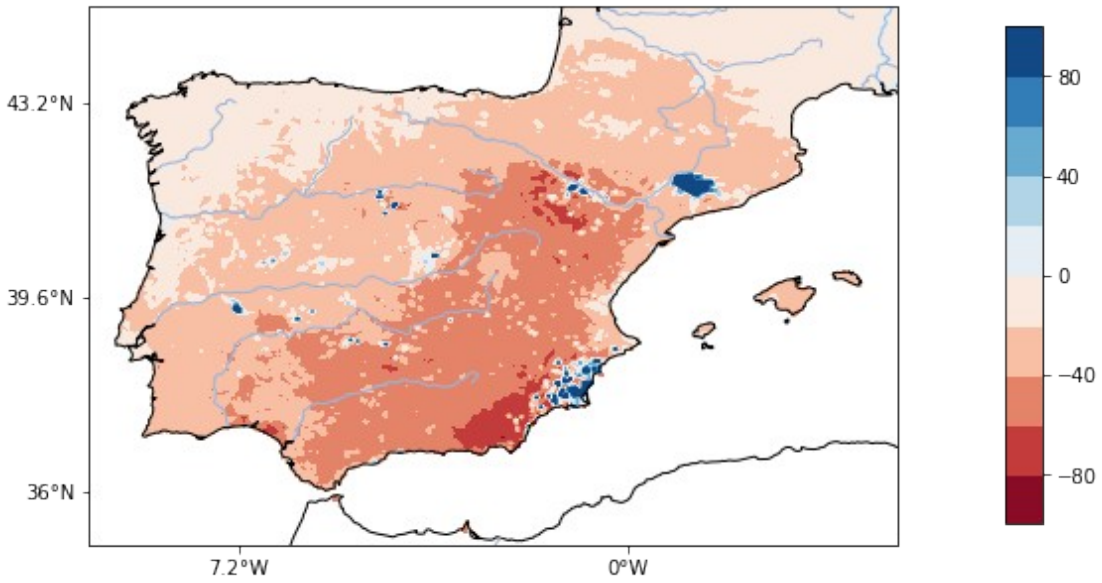
```
3601.604518890381
-99.29529635111491
100.0
-99.29529635111491
```

Ahora podemos visualizar el resultado.

```
fig = plt.figure(figsize=(9, 5))
map_projection = ccrs.PlateCarree()
ax = plt.axes(projection=map_projection)
clevs = [-1.0, -0.7, -0.45, -0.16, 0.5, 0.7]
im = ax.contourf(lons, lats, data, levels=10, cmap='RdBu',
transform=map_projection)
ax.add_feature(cfeature.COASTLINE)
ax.add_feature(cfeature.RIVERS)

ax.set_xticks(np.linspace(-180, 180, 51), crs=map_projection) #puede
ser necesario modificar
ax.set_yticks(np.linspace(-90, 90, 51), crs=map_projection)   #puede
```

```
ser necesario modificar
lon_formatter = LongitudeFormatter(zero_direction_label=True)
lat_formatter = LatitudeFormatter()
ax.xaxis.set_major_formatter(lon_formatter)
ax.yaxis.set_major_formatter(lat_formatter)

dspace = 0.01 # distance between the main image and the colorbar.
dwidth = 0.03 # width of the colorbar.
cax = fig.add_axes([ax.get_position().x1 + dspace,
                    ax.get_position().y0,  dwidth,
                    ax.get_position().height])
plt.colorbar(im, cax=cax)

ax.set_extent((-10, 5, 35, 45), crs=ccrs.PlateCarree()) #cambiar a la
zona de interés
plt.show()
```

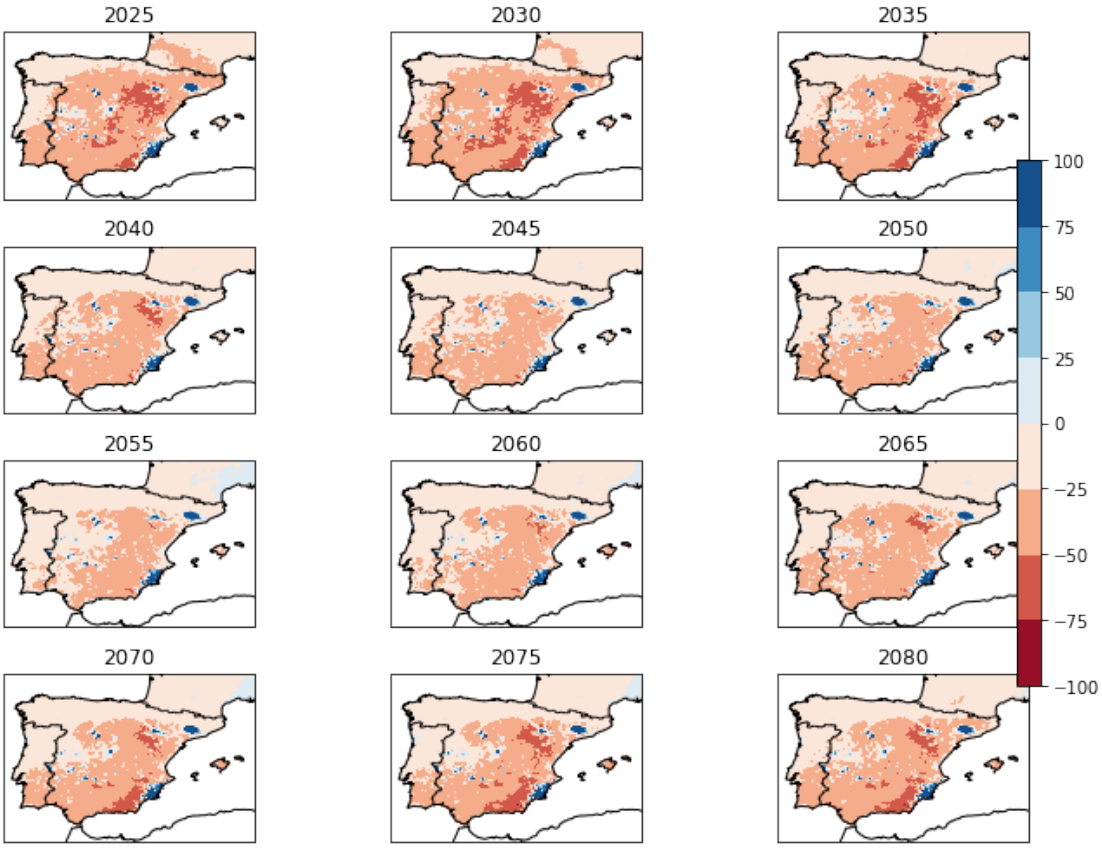
```
/usr/local/lib/python3.7/dist-packages/cartopy/io/__init__.py:241:
DownloadWarning: Downloading:
https://naciscdn.org/naturalearth/10m/physical/ne_10m_coastline.zip
  warnings.warn('Downloading: {}'.format(url), DownloadWarning)
/usr/local/lib/python3.7/dist-packages/cartopy/io/__init__.py:241:
DownloadWarning: Downloading:
https://naciscdn.org/naturalearth/10m/physical/ne_10m_rivers_lake_cent
erlines.zip
  warnings.warn('Downloading: {}'.format(url), DownloadWarning)
```



El mapa anterior nos muestra los resultados para un modelo y un único periodo futuro. Sería interesante ver, aunque fuera solo para un modelo, la evolución a lo largo del tiempo que nos indica el mismo.

```
fig, axs = plt.subplots(ncols=3,nrows=4,figsize=(10,7),
                        subplot_kw={'projection': ccrs.PlateCarree()})
#fig.tight_layout()
periods = np.arange(1,13)
periods_labels=[2025,2030,2035,2040,2045,2050,
                2055,2060,2065,2070,2075,2080]
map_projection = ccrs.PlateCarree()
for m in range(len(periods)):
    i = int(m/3)  #m % 4
    j = m % 3 #int(m/4)
    #print(i,j)

    ax = axs[i,j]
    im = ax.contourf(lons, lats, big_array_mean[m],cmap='RdBu')
    ax.add_feature(cfeature.COASTLINE)
    ax.add_feature(cfeature.BORDERS)
    ax.title.set_text(periods_labels[m])
    ax.set_xticks([])
    ax.set_yticks([])
    ax.set_extent((-10, 5, 35, 45), crs=ccrs.PlateCarree())

dspace = 0.01 # distance between the main image and the colorbar.
dwidth = 0.03 # width of the colorbar.
cax = fig.add_axes([0.9, 0.2, 0.02, 0.6])
plt.colorbar(im, cax=cax)

plt.tight_layout()
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:27:
UserWarning: This figure includes Axes that are not compatible with
tight_layout, so results might be incorrect.
/usr/local/lib/python3.7/dist-packages/cartopy/io/__init__.py:241:
DownloadWarning: Downloading:
https://naciscdn.org/naturalearth/10m/cultural/ne_10m_admin_0_boundary
_lines_land.zip
  warnings.warn('Downloading: {}'.format(url), DownloadWarning)
```



Selección de una ubicación concreta

Los datos están en una rejilla un poco especial, ya que no está alineada con las latitudes y longitudes. Una posible solución para ver que cuadrícula de la misma nos interesa es calcular la distancia de un punto de interés a los puntos de la rejilla más cercanos.

```
lon_de_interes=-5.667529868098542
lat_de_interes=40.956530527184086

distancias_a_pto=((lons-lon_de_interes)**2+(lats-
lat_de_interes)**2)**0.5
print(distancias_a_pto.shape)

positionY,positionX=np.where(distancias_a_pto==np.min(distancias_a_pto
))

positions_of_interes=(positionY.tolist()+positionX.tolist())
print(positions_of_interes)

#comprobamos
print(lons[positions_of_interes[0]][positions_of_interes[1]])
print(lats[positions_of_interes[0]][positions_of_interes[1]])
```

```
#thanks to
#https://es.wikipedia.org/wiki/Distancia_euclidiana
#https://thispointer.com/find-the-index-of-a-value-in-numpy-array/
#https://numpy.org/doc/stable/reference/generated/numpy.ndarray.tolist
.html
```

```
(950, 1000)
[675, 101]
-5.662493677836304
40.97211793907815
```

Una vez que sabemos como extraer los datos de un punto, podemos aplicar esto con el siguiente código. Verás que hemos obtenido unos datos que sobretodo son "nan" (not a numbers) eso es porque los bordes de mi zona de interés están sobre el mar, y para el mar el archivo no aporta valores.

```
dataDIR = 'GFDL-ESM2M_rcp2p6_mHM_streamflow_mean-monthly_2011-
2095_'+'JAN'+'_v1.nc' #we are already in the right folder. add the
path if needed.
DS = xr.open_dataset(dataDIR)
relative_changes=DS.relative_change.values

print(relative_changes.shape)
relative_changes=np.squeeze(relative_changes)
print(relative_changes.shape)

relative_change_my_location=relative_changes[:,500,500]
print(relative_changes)
```

```
(12, 1, 950, 1000)
(12, 950, 1000)
[[[nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  ...
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]]

 [[nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  ...
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]]

 [[nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
```

```
  [nan nan nan ... nan nan nan]
  ...
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]]

 ...

 [[nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  ...
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]]

 [[nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  ...
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]]

 [[nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  ...
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]
  [nan nan nan ... nan nan nan]]]
```

Análisis de varios modelos en mi ubicación de interés.

A continuación aplicamos un bucle para repetir esta operación en todas las combinaciones de modelos, escenarios, etc. En Python, los bucles se pueder realizar con `for`. El código que se repite dentro del mismo está desplazado o "indentado" hacia la derecha. De este modo, se pueden emplear bucles dentro de bucles. Nosotros vamos a hacer un bucle que recorra todos los modelos, dentro del cual habra otro que recorra los diferentes modelos hidrológicos, y por último, dentro de este recorreremos los datos de cada mes.

```
summary_of_data=np.zeros((12,len(gcm_models)*len(hidrological_models)*
len(months)))
list_of_models_and_scenarios=[]
i=0
for g in gcm_models:
    for h in hidrological_models:
        for m in months:
```

```
            print(i,g,h,m)
            dataDIR = g +'_rcp2p6_'+h+'_streamflow_mean-monthly_2011-
2095_'+m+'_v1.nc' #we are already in the right folder. add the path if
needed.
            DS = xr.open_dataset(dataDIR)

relative_changes=DS.relative_change.values[:,:,positions_of_interes[0]
,positions_of_interes[1]]
            #print(relative_changes)
            summary_of_data[:,i]=np.squeeze(relative_changes)
            list_of_models_and_scenarios.append(g+'_'+h+'_'+m)
            DS.close()
            i=i+1
```

```
0 GFDL-ESM2M mHM JAN
1 GFDL-ESM2M mHM FEB
2 GFDL-ESM2M mHM MAR
3 GFDL-ESM2M mHM APR
4 GFDL-ESM2M mHM MAY
5 GFDL-ESM2M mHM JUN
6 GFDL-ESM2M mHM JUL
7 GFDL-ESM2M mHM AUG
8 GFDL-ESM2M mHM SEP
9 GFDL-ESM2M mHM OCT
10 GFDL-ESM2M mHM NOV
11 GFDL-ESM2M mHM DEC
12 GFDL-ESM2M noah-mp JAN
13 GFDL-ESM2M noah-mp FEB
14 GFDL-ESM2M noah-mp MAR
15 GFDL-ESM2M noah-mp APR
16 GFDL-ESM2M noah-mp MAY
17 GFDL-ESM2M noah-mp JUN
18 GFDL-ESM2M noah-mp JUL
19 GFDL-ESM2M noah-mp AUG
20 GFDL-ESM2M noah-mp SEP
21 GFDL-ESM2M noah-mp OCT
22 GFDL-ESM2M noah-mp NOV
23 GFDL-ESM2M noah-mp DEC
24 GFDL-ESM2M PCRGLOBWB JAN
25 GFDL-ESM2M PCRGLOBWB FEB
26 GFDL-ESM2M PCRGLOBWB MAR
27 GFDL-ESM2M PCRGLOBWB APR
28 GFDL-ESM2M PCRGLOBWB MAY
29 GFDL-ESM2M PCRGLOBWB JUN
30 GFDL-ESM2M PCRGLOBWB JUL
31 GFDL-ESM2M PCRGLOBWB AUG
32 GFDL-ESM2M PCRGLOBWB SEP
33 GFDL-ESM2M PCRGLOBWB OCT
34 GFDL-ESM2M PCRGLOBWB NOV
35 GFDL-ESM2M PCRGLOBWB DEC
```

```
36 GFDL-ESM2M VIC JAN
37 GFDL-ESM2M VIC FEB
38 GFDL-ESM2M VIC MAR
39 GFDL-ESM2M VIC APR
40 GFDL-ESM2M VIC MAY
41 GFDL-ESM2M VIC JUN
42 GFDL-ESM2M VIC JUL
43 GFDL-ESM2M VIC AUG
44 GFDL-ESM2M VIC SEP
45 GFDL-ESM2M VIC OCT
46 GFDL-ESM2M VIC NOV
47 GFDL-ESM2M VIC DEC
```

A continuacion vamos a analizar todos los modelos a los que tenemos acceso y . Como los datos los tenemos ya en una matriz, la vamos a guardar primero como un marco de datos o "data frame" para poder consultarlo más fácilmente. Un dataframe viene a ser una pestaña de una hoja de cálculo (p. ej. Microsoft Excel), y se organiza en filas y columnas.

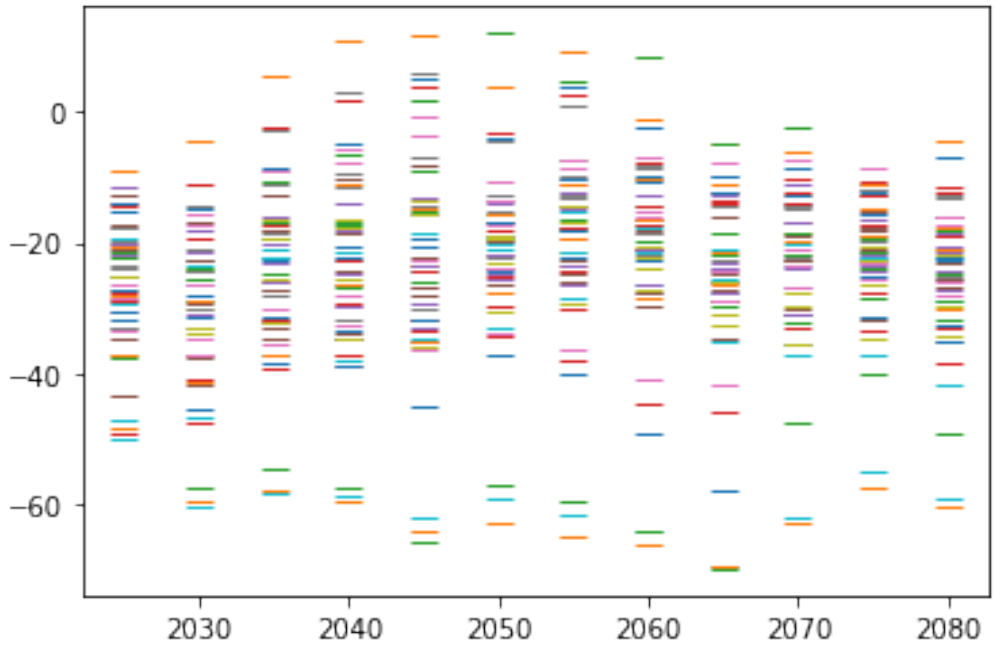
```
dataframe01=pd.DataFrame(data=summary_of_data,

index=(2025,2030,2035,2040,2045,2050,2055,2060,2065,2070,2075,2080),
                          columns=list_of_models_and_scenarios)

print(dataframe01.size)
```

```
576
```

Una vez que tenemos los resultados almacenados, podemos fácilmente generar un gráfico con los posibles cambios de todos los modelos y escenarios.

```
dataframe01.plot(legend=False,linestyle='None', marker='_', markersize
= 10)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f94fed0de50>
```



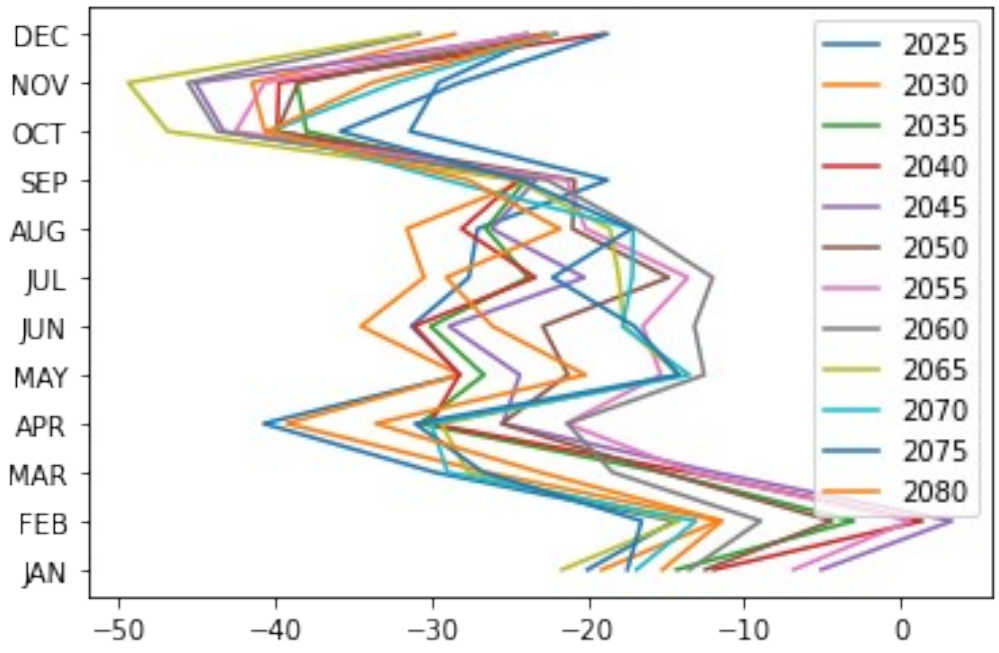
Otro posible análisis es el promedio de los cambios de cada mes. Como verás, la señal climatíca que aporta este modelo es marcadamente estacional y no un cambio medio para todos los meses.

```
cambios_mensuales=np.zeros((12,12))
i=0
for m in months:
    list_of_selected_columns=[]
    for g in gcm_models:
        for h in hidrological_models:
            list_of_selected_columns.append(g+'_'+h+'_'+m)
    print(list_of_selected_columns)
    print()
    print("Extraccion")
    dataframe2=dataframe01[list_of_selected_columns]
    cambios_mensuales[i]=dataframe2.mean(axis=1).values
    i=i+1
print(cambios_mensuales)

plt.plot(cambios_mensuales,months)
plt.legend(periods_labels)
```

```
['GFDL-ESM2M_mHM_JAN', 'GFDL-ESM2M_noah-mp_JAN', 'GFDL-
ESM2M_PCRGLOBWB_JAN', 'GFDL-ESM2M_VIC_JAN']

Extraccion
['GFDL-ESM2M_mHM_FEB', 'GFDL-ESM2M_noah-mp_FEB', 'GFDL-
ESM2M_PCRGLOBWB_FEB', 'GFDL-ESM2M_VIC_FEB']

Extraccion
```

```
['GFDL-ESM2M_mHM_MAR', 'GFDL-ESM2M_noah-mp_MAR', 'GFDL-
ESM2M_PCRGLOBWB_MAR', 'GFDL-ESM2M_VIC_MAR']

Extraccion
['GFDL-ESM2M_mHM_APR', 'GFDL-ESM2M_noah-mp_APR', 'GFDL-
ESM2M_PCRGLOBWB_APR', 'GFDL-ESM2M_VIC_APR']

Extraccion
['GFDL-ESM2M_mHM_MAY', 'GFDL-ESM2M_noah-mp_MAY', 'GFDL-
ESM2M_PCRGLOBWB_MAY', 'GFDL-ESM2M_VIC_MAY']

Extraccion
['GFDL-ESM2M_mHM_JUN', 'GFDL-ESM2M_noah-mp_JUN', 'GFDL-
ESM2M_PCRGLOBWB_JUN', 'GFDL-ESM2M_VIC_JUN']

Extraccion
['GFDL-ESM2M_mHM_JUL', 'GFDL-ESM2M_noah-mp_JUL', 'GFDL-
ESM2M_PCRGLOBWB_JUL', 'GFDL-ESM2M_VIC_JUL']

Extraccion
['GFDL-ESM2M_mHM_AUG', 'GFDL-ESM2M_noah-mp_AUG', 'GFDL-
ESM2M_PCRGLOBWB_AUG', 'GFDL-ESM2M_VIC_AUG']

Extraccion
['GFDL-ESM2M_mHM_SEP', 'GFDL-ESM2M_noah-mp_SEP', 'GFDL-
ESM2M_PCRGLOBWB_SEP', 'GFDL-ESM2M_VIC_SEP']

Extraccion
['GFDL-ESM2M_mHM_OCT', 'GFDL-ESM2M_noah-mp_OCT', 'GFDL-
ESM2M_PCRGLOBWB_OCT', 'GFDL-ESM2M_VIC_OCT']

Extraccion
['GFDL-ESM2M_mHM_NOV', 'GFDL-ESM2M_noah-mp_NOV', 'GFDL-
ESM2M_PCRGLOBWB_NOV', 'GFDL-ESM2M_VIC_NOV']

Extraccion
['GFDL-ESM2M_mHM_DEC', 'GFDL-ESM2M_noah-mp_DEC', 'GFDL-
ESM2M_PCRGLOBWB_DEC', 'GFDL-ESM2M_VIC_DEC']

Extraccion
[[-20.01441836 -19.09236479 -14.31347871 -11.936849    -5.06245659
  -12.47330081  -6.81879818 -13.49858505 -21.63767743 -16.87799001
  -17.46902728 -15.2260834 ]
 [-14.45081544 -11.5943079   -2.97215647   1.41099763   3.30875635
   -4.35619903   0.63732272  -8.92971969 -14.11941969 -13.07371151
  -16.54628825 -11.39955723]
 [-29.79976988 -26.47948933 -15.22054625 -13.55358779 -11.93175125
  -15.26122856 -13.27536952 -18.42128658 -27.67558098 -28.97270298
  -26.69954109 -23.05523682]
```

```
 [-40.74339581 -39.27585077 -30.72125816 -30.08869243 -25.4297713
  -25.51163435 -21.35869336 -21.29662752 -29.37354422 -30.02856255
  -31.04863071 -33.57326317]
 [-28.1376729  -28.15552568 -26.6424346  -28.21335697 -24.3538146
  -21.26708031 -15.31031275 -12.54475772 -14.08626091 -13.49853456
  -14.17188239 -20.13926554]
 [-31.27968979 -34.51149416 -30.05604982 -31.12492371 -28.88526773
  -22.8847518  -16.48021555 -13.16964054 -17.69517875 -17.78079677
  -17.01370835 -26.10091352]
 [-27.60385752 -30.4694519  -23.8061583  -23.35952997 -20.21973562
  -14.82404923 -13.5847975  -12.00352645 -18.0010314  -17.14869952
  -22.24801755 -29.05836773]
 [-27.05476141 -31.57644749 -26.5162673  -28.07956982 -26.15248489
  -20.97929764 -20.12952614 -16.86907482 -18.57115459 -17.09457469
  -17.19276667 -21.77370548]
 [-18.73502779 -23.83888149 -23.9640255  -24.47477531 -23.31320381
  -20.83659172 -21.32788324 -22.58536768 -24.55301809 -29.04168224
  -24.29050446 -27.61813354]
 [-31.37228608 -40.66266012 -38.00314283 -39.89635372 -43.3002038
  -39.88806963 -42.54266691 -43.68905616 -46.87705302 -40.3636477
  -35.84297061 -40.5100913 ]
 [-29.55573988 -41.48965359 -38.60298252 -39.76083374 -45.12319183
  -38.53584528 -40.68501139 -45.59825897 -49.41681623 -32.28991032
  -28.21138668 -33.81498003]
 [-22.69852924 -28.5073266  -23.92265654 -18.97194433 -22.74641585
  -22.03702879 -23.85557151 -30.74798298 -31.01723766 -22.05758166
  -18.7726922  -22.36697125]]

<matplotlib.legend.Legend at 0x7f94ff567b90>
```



Tratamiento de series de caudales históricos

Una vez que hemos visto como manejar las proyecciones de cambio que nos aporta el servicio de Cambio Climático de Copernicus, vamos a var como estas se pueden aplicar a datos históricos que representen el clima pasado. Para ello es preciso adquirir datos históricos de un aforo.

En España existen diferentes fuentes para ello. Por ejemplo, se puede consultar el Sistema de Información Geográfica de MAPAMA, o los anuarios de aforos de CEDEX

En este caso hemos descargado los aforos para nuestra ubicación y los hemos tratado previamente para eliminar valores anómalos, negativos, huecos, etc. Los datos están en la columna C del archivo CaudalesEjemplo01.xlsx. Para leer el mismo se procede de la siguiente manera.

```
path_to_file=''
df = pd.read_excel ((path_to_file+'CaudalesEjemplo01.xlsx'),
                    sheet_name='Hoja1',
                    skiprows=[0],
                    usecols = "A,C")#las columnas que quiero
print (df)
```

```
           Fecha                        Caudal corregido (m3/s)
0     1970-01-01                                            NaN
1     1970-01-02                                            NaN
2     1970-01-03                                            NaN
3     1970-01-04                                            NaN
4     1970-01-05                                            NaN
...          ...                                            ...
17436 2017-09-27                                       0.579148
17437 2017-09-28                                              0
17438 2017-09-29                                     3.7037e-05
17439 2017-09-30                                       0.346815
17440        NaT  Esta es una fila que tiene que ser eliminada

[17441 rows x 2 columns]
```

```
print (df.tail(10)) #me muestra si tengo que quitar  filas al final
```

```
           Fecha                        Caudal corregido (m3/s)
17431 2017-09-22                                       0.231852
17432 2017-09-23                                       0.231963
17433 2017-09-24                                       0.346889
17434 2017-09-25                                    0.000111111
17435 2017-09-26                                       0.231704
17436 2017-09-27                                       0.579148
17437 2017-09-28                                              0
17438 2017-09-29                                     3.7037e-05
17439 2017-09-30                                       0.346815
17440        NaT  Esta es una fila que tiene que ser eliminada
```

```
df=df.drop(range(17440,17441,1),axis=0) #quito una fila que no valen
abajo del excel
print (df.tail(10)) #listo
```

```
           Fecha Caudal corregido (m3/s)
17430 2017-09-21              3.7037e-05
17431 2017-09-22                0.231852
17432 2017-09-23                0.231963
17433 2017-09-24                0.346889
17434 2017-09-25             0.000111111
17435 2017-09-26                0.231704
17436 2017-09-27                0.579148
17437 2017-09-28                       0
17438 2017-09-29              3.7037e-05
17439 2017-09-30                0.346815
```

```
print (df.columns) #el nombre de las columnas puede ser muy complicado
```

```
Index(['Fecha', 'Caudal corregido (m3/s)'], dtype='object')
```

```
#damos nombre a las columnas
df.columns = ['Fecha','Caudal']
print(df)
```

```
           Fecha      Caudal
0     1970-01-01         NaN
1     1970-01-02         NaN
2     1970-01-03         NaN
3     1970-01-04         NaN
4     1970-01-05         NaN
...          ...         ...
17435 2017-09-26    0.231704
17436 2017-09-27    0.579148
17437 2017-09-28           0
17438 2017-09-29  3.7037e-05
17439 2017-09-30    0.346815

[17440 rows x 2 columns]
```

```
#realmente los valores almacenados en la columna "Caudal" son textos.
Hay que convertirlos en números
df.Caudal=pd.to_numeric(df.Caudal,errors='coerce')

#generar un indice que se pueda manejar como fechas
df.set_index('Fecha', inplace=True)
df.index = pd.to_datetime(df.index)
print(df)
```

```
              Caudal
Fecha
1970-01-01       NaN
1970-01-02       NaN
```

```
1970-01-03       NaN
1970-01-04       NaN
1970-01-05       NaN
...              ...
2017-09-26  0.231704
2017-09-27  0.579148
2017-09-28  0.000000
2017-09-29  0.000037
2017-09-30  0.346815

[17440 rows x 1 columns]
```

Con todo lo anteior generamos una tabla en la que podremos aplicar nuestros cambios relativos. Esto lo realizamos del siguiente modo:

```
#recuperar el cambio a aplicar a cada valor y calcular nuevos caudales
df['Mes']=df.index.month
df['Cambio2025']=cambios_mensuales[0][df.Mes-1]
df['Caudal2025']=df['Caudal']*(100+df['Cambio2025'])/100
print(df)
```

```
              Caudal  Mes  Cambio2025  Caudal2025
Fecha
1970-01-01       NaN    1  -20.014418         NaN
1970-01-02       NaN    1  -20.014418         NaN
1970-01-03       NaN    1  -20.014418         NaN
1970-01-04       NaN    1  -20.014418         NaN
1970-01-05       NaN    1  -20.014418         NaN
...              ...  ...         ...         ...
2017-09-26  0.231704    9  -21.637677    0.181568
2017-09-27  0.579148    9  -21.637677    0.453834
2017-09-28  0.000000    9  -21.637677    0.000000
2017-09-29  0.000037    9  -21.637677    0.000029
2017-09-30  0.346815    9  -21.637677    0.271772

[17440 rows x 4 columns]
```

Declaración de descargo de responsabilidad

Este documento ha sido producido con el apoyo financiero de la Fundación Biodiversidad, del Ministerio para la Transición Ecológica y el Reto Demográfico. Las opiniones expresadas en este documento no pueden considerarse de ninguna manera como un reflejo de la opinión oficial de dichas instituciones. TECNALIA, como autor de esta herramienta, no se hace responsable del uso que se le dé y no admite ningún tipo de responsabilidad por los resultados que se obtengan de su uso ni por posibles daños, directos o indirectos, que pudieran originarse como consecuencia de la aplicación práctica de dichos resultados.