|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Python function | Training script | Objectif | Explication | Example | Modules needed |
| information\_netCDF(files,time='YES',lat='YES',lon='YES',output='Screen') | example\_information\_netCDF.py | This function prints the short name of variables and their dimensions. The dimensions of the field must be noted with time, lon and lat to obtain the information about them | files = put here the path and the name of the netCDF file time = put 'YES' if you want information about the time dimension or anything else if not lat = put 'YES' if you want information about the lat dimension or anything else if not lon = put 'YES' if you want information about the lon dimension or anything else if not output = put 'Screen' if you want the information printed on the screen or  the path and the txt file if you want the information to be written into a txt file | This function is more for internal use; it prints information on netCDF file: short name of variables; lat and lon; and time binds | import xarray import sys |
| min\_mean\_max\_figures(file, var, attributes='YES',x='lon', y='lat', cmap='BrBG\_r', levels=22) | example\_min\_mean\_max\_figures.py | The function will plot thee figures showing the minimum, maximum and the mean values over the entire period | files = put here the path and the name of the netCDF file var=put here the name of the variable you want to plot (ex. 'tas') attributes = put 'YES' if you want to see the attributes of the variable x= put the name of the dimension ox y= put the name of the dimension 0y cmap= put the name of the colour scale to use  levels= put the number of colours to use. | This function is more for internal use; it perits to have a general idea about the 3D netcdf file by showing on maps what are the minimumm, maxium and mean values over time in each grid point. | import xarray |
| sel\_Period\_Region(file,first\_date, last\_date, lat\_bottom, lat\_top, lon\_left, lon\_right, show\_regInfo='YES', save\_nerCDF='NO') | example1\_sel\_Period\_Region.py example2\_sel\_Period\_Region.py | This function will open a netcDF file, select data situated into a rectangle defined by the closest points to lat\_bottom, lat\_top, lon\_left, lon\_right, and the time period and save the new file in netCDF.  The function supposes that the file has the spatial dimensions noted with lat and lon. | file = put here the path and the name of the original netCDF file lat\_bottom, lat\_top, lon\_left, lon\_right= put here the approximative coordinates of the region show\_newPer= put 'YES' if you want to verify the lat and long selected first\_date = date from which the selection begins ; ex. '2000-02-15' last\_date = last date in the selected period; ex. '2002-10-10' save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User requests netCDF files for CMIP5 projections for precipitation over the Prairie region (more exactly, over the region defined by the longitudes 115 W to 95 W and the latitudes 49N to 55 N), from 2020 to 2100 for all CMIP5 models. The output is netCDF. example1 is for one model; example 2 is for several models; | import xarray |
| sel\_and\_mean\_Region(file,var,lat\_bottom, lat\_top, lon\_left, lon\_right, show\_regInfo='YES', save\_CSV='NO') | example1\_sel\_and\_mean\_Region.py example2\_sel\_and\_mean\_Region.py | This function will open an netcDF file, select data situated into a rectangle defined by the closest points to lat\_bottom, lat\_top, lon\_left, lon\_right, make an area-weighted average of gridded spatial data and save the new file.  The function supposes that the file has the spatial dimensions noted with lat and lon. | file = put here the path and the name of the original netCDF file var = put here the name of the variable, ex. 'tas' lat\_bottom, lat\_top, lon\_left, lon\_right= put here the approximative coordinates of the region show\_newPer= put 'YES' if you want to verify the lat and lon selected save\_CSV= if you want to save the file put here the path and the name of the CSV file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User requests CMIP5 projections for the AVERAGE temperature value over the Maritime region, from 2020 to 2070, under one particular model or for all CMIP5 models, each model saved separatly in CSV. (the input is netCDF; the output is CSV); example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| sel\_and\_mean\_Region\_liste(input,liste,varName,lat\_bottom, lat\_top, lon\_left, lon\_right, show\_regInfo='YES', save\_CSV='NO') | example\_sel\_and\_mean\_Region\_liste.py | This function will open several netcDF files, for each of them will select data situated into a rectangle defined by the closest points to lat\_bottom, lat\_top, lon\_left, lon\_right, make an area-weighted average of gridded spatial data and save the information from all models into one CSV file. The function supposes that the file has the spatial dimensions noted with lat and lon. | input = put here the path to the netCDF files liste = put here the liste with the names of oll netCDF files you want varName = put here the name of the variable, ex. 'tas' lat\_bottom, lat\_top, lon\_left, lon\_right= put here the approximative coordinates of the region show\_newPer= put 'YES' if you want to verify the lat and lon selected save\_CSV= if you want to save the file put here the path and the name of the CSV file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User requests CMIP5 projections for temperature or other variables AVERAGE value over the Prairie region (more exactly, average over the region defined by the longitudes 115 W to 95 W and the latitudes 49N to 55 N), from 2020 to 2100, for all CMIP5 models, (the input is netCDF; the output is ONE CSV file with each model on a column) | import time import numpy as np import pandas as pd import xarray import glob, os |
| change\_to\_mm\_per\_day(file, var, save='NO', figure='YES') | example1\_change\_to\_mm\_per\_day.py example2\_change\_to\_mm\_per\_day.py | This function transforms kg m-2 s-1 in mm/day for a netCDF file and save the new file as netCDF | file = put here the path and the name of the netCDF file in kg m-2 s-1 var = put here the name of the variable you want to change from kg m-2 s-1 to mm/day  save = if you want to save the file put here the path and the name of the netCDF file in mm/day; if you don't want to save it, put 'NO' and use the file locally for other operations figure = put 'YES' if want to see a figure with the mean over the entire period. | No longer useful. Data already in mm/day. User requests precipitation values in netCDF file to be converted from kg/sm2 to mm per day. | import numpy as np import pandas as pd import xarray import matplotlib.pyplot as plt |
| Kelvin\_to\_Celsius(file, var, save='NO', figure='YES') | example1\_Kelvin\_to\_Celsius.py example2\_Kelvin\_to\_Celsius.py | This function transforms Kelvin in Celsius for a netCDF file and save the new file as netCDF. | file = put here the path and the name of the netCDF file in Kelvin var = put here the name of the variable you want to change from Kelvin to Celsius save = if you want to save the file put here the path and the name of the netCDF file in Celsius; if you don't want to save it, put 'NO' and use the file locally for other operations figure = put 'YES' if want to see a figure with the mean over the entire period. | No longer useful. Data already in Celsius. User requests temperature values in netCDF file to be converted from K to C | import numpy as np import pandas as pd import xarray import matplotlib.pyplot as plt |
| sel\_Region(file, lat\_bottom, lat\_top, lon\_left, lon\_right, show\_regInfo='YES', save\_nerCDF='NO') | example1\_sel\_Region.py example2\_sel\_Region.py | This function will open a netcDF file, select data situated into a rectangle defined by the closest points to lat\_bottom, lat\_top, lon\_left, lon\_right, and save the new file in netCDF. The function supposes that the file has the spatial dimensions noted with lat and lon. | file = put here the path and the name of the original netCDF file lat\_bottom, lat\_top, lon\_left, lon\_right= put here the approximative coordinates of the region show\_newPer= put 'YES' if you want to verify the lat and long selected save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User desires the spatial confinement of the Baffin Islands (not point but region) on the netCDF file from a netCDF file of Canada. They provide the coordinates in lat and lon for the region. The input is netCDF and the output is netcDF. example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| selperiod\_date(file,first\_date, last\_date, show\_newPer='YES', save\_nerCDF='NO') | example1\_selperiod\_date.py example2\_selperiod\_date.py | This function will open an netcDF file, select all time steps from the date corresponding to first\_date to the date corresponding to the last\_date and save the new file in netCDF | file = put here the path and the name of the original netCDF file first\_date = date from which the selection begins ; ex. '2000-02-15' last\_date = last date in the selected period; ex. '2002-10-10' show\_newPer= put 'YES' if you want to verify the years selected save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants netCDF files with data between specific dates, for example from 2007-03-24 to 2018-04-25. The input is netCDF and the output is netcDF. example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| selperiod\_years(file,first\_year, last\_year, show\_newPer='YES', save\_nerCDF='NO') | excample1\_selperiod\_years.py excample2\_selperiod\_years.py | This function will open a netcDF file, select all time steps from the date corresponding to first\_year to the date corresponding to the last year and save the new file in netCDF | file = put here the path and the name of the original netCDF file first\_year = year from which the selection begins ; ex. '1982' last\_year = last year in the selected period; ex. '2000' show\_newPer= put 'YES' if you want to verify the years selected save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants netCDF files with data for a specific range of years, for example from 2004 - 2010.The input is netCDF and the output is netcDF. example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| spatial\_mean(file,var, save\_CSV='NO') | example1\_spatial\_mean.py example2\_spatial\_mean.py | This function will open a netcDF file, make a area-weighted averages of gridded spatial data and save the new file. The function supposes that the file has the spatial dimensions noted with lat and lon. | file = put here the path and the name of the original netCDF file var = put here the name of the variable, ex. 'tas' save\_CSV= if you want to save the file put here the path and the name of the CSV file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | This script is for internal use and can be useful when more then one function is nesessary. Example: user wants climate projections for RCP 8.5, for temperature, annual means, average over the Prairie region (more exactly, average value over the region defined by the longitudes 115 W to 95 W and the latitudes 49N to 55 N).You will use first other functions (1) to select the period and the region, (2) to do the annual mean and save the new data as netCDF (this data will be only netCDF annual values over the period and over the region desired by user ). You can use next this function (spatial\_mean) to compute the spatial mean for the netcDF previously obtained files. The input will be in netCD, the output will be CSV (one CSV per model). example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| spatial\_mean\_liste(input,liste, var, save\_CSV='NO') | example\_spatial\_mean\_liste.py | This function will open all netcDF files, make area-weighted averages of gridded spatial data for each file and save them into one csv file. The function supposes that the file has the spatial dimensions noted with lat and lon. | input = put here the path to the netCDF files  liste = put here the liste with the names of all netCDF files you want  var = put here the name of the variable, ex. 'tas'  save\_CSV= if you want to save the file put here the path and the name of the CSV file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | This script is for internal use and can be useful when more then one function is nesessary. Example: user wants climate projections for RCP 8.5, for temperature, annual means, average over the Prairie region (more exactly, average value over the region defined by the longitudes 115 W to 95 W and the latitudes 49N to 55 N).You will use first other functions (1) to select the period and the region, (2) to do the annual mean and save the new data as netCDF (this data will be only netCDF annual values over the period and over the region desired by user ). You can use next this function (spatial\_mean\_list) to compute the spatial mean for the netcDF previously obtained files. The input will be in netCD, the output will be ONE CSV with each model on one column. | import numpy as np import pandas as pd import xarray |
| selPoint(file, var, latitude, longitude, save\_nerCDF='NO',save\_CSV='NO') | example1\_selPoint.py example2\_selPoint.py | This function will open a netcDF file, select data for the grid point situated closest   to latitude and longitude indicated and save the new file in netCDF or CSV.  The function supposes that the file has the spatial dimensions noted with lat and lon. | file = put here the path and the name of the original netCDF file  var = put here the name of the variable, ex. 'tas'  latitude, longitude= put here the approximative coordinates of the point  save\_CSV= if you want to save the file put here the path and the name of the CSV file to save; if you don't want to save it, put 'NO' and use the file locally for other operations  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User desires precipitation projections or historical simulated values (entire period) in High Park, Toronto (exact location) from a CMIP5 model or several models. The input will be 3D netCDF, the output will be CSV or netCDF(one file per model). example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| selPoint\_liste(input, liste, var, latitude, longitude, save\_CSV='NO') | example\_selPoint\_liste.py | This function will open several netcDF files, for each of them will select data for  the grid point situated closest to latitude and longitude indicated and save the information from all models in one csv  input = put here the path to the netCDF files | input = put here the path to the netCDF files  liste = put here the liste with the names of oll netCDF files you want  var = put here the name of the variable, ex. 'tas'  latitude, longitude= put here the approximative coordinates of the point  save\_CSV= if you want to save the file put here the path and the name of the CSV file to save; if you don't want to save it, put 'NO' and use the file locally for other operations | User wants data for Saskatoon International Airport for the entire period, for a variety of netCDF files (ie. different models). The input will be 3D netCDF files, the output will be in a single CSV file. | import numpy as np import pandas as pd import xarray |
| annual\_max\_3D(file, show\_newPer='YES', save\_nerCDF='NO') | example1\_annual\_max\_3D.py example2\_annual\_max\_3D.py | This function will open an 3D netcDF file, compute the time maximum for each year and save the new 3D file in netCDF. The date indicated for each year is the last time step of the corresponding year. | file = put here the path and the name of the original netCDF file  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants to get the maximum temperature for every year in a new netCDF file. The input will be netCDF, the output will be netCDF(one file per model). example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| annual\_min\_3D(file, show\_newPer='YES', save\_nerCDF='NO') | example1\_annual\_min\_3D.py example2\_annual\_min\_3D.py | This function will open an 3D netcDF file, compute the time minimum for each year and save the new 3D file in netCDF.The date indicated for each year is the last time step of the corresponding year. | file = put here the path and the name of the original netCDF file  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants to get the minimum temperature for every year in a new netCDF file.The input will be netCDF, the output will be netCDF(one file per model). example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| annual\_mean\_3D(file, show\_newPer='YES', save\_nerCDF='NO') | example1\_annual\_mean\_3D.py example2\_annual\_mean\_3D.py | This function will open an 3D netcDF file, compute the time mean for each year and save the new file in netCDF.The date indicated for each year is the last time step of the corresponding year. | file = put here the path and the name of the original netCDF file  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save; if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants to get the average temperature for every year in a new netCDF file.The input will be netCDF, the output will be netCDF(one file per model). example1 is for one model; example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| monthly\_mean\_3D(file, show\_newPer='YES', save\_nerCDF='NO') | example1\_monthly\_mean\_3D.py example2\_monthly\_mean\_3D.py | This function will open an 3D netcDF file, compute the time mean for each month and save the new file in netCDF. The date indicated for each year is the first time step of the corresponding month. | file = put here the path and the name of the original netCDF file  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants to get the average precipitation for every month in a new netCDF file.The input will be netCDF, the output will be netCDF(one file per model). example1 is for one model | import numpy as np import pandas as pd import xarray |
| seasonal\_mean\_3D(file, show\_newPer='YES', save\_nerCDF='NO') | example1\_seasonal\_mean\_3D.py example2\_seasonal\_mean\_3D.py | This function will open an 3D netcDF file, compute the time mean for each season and save the new file in netCDF.  The date indicated for each season is the last time step of the corresponding season. | file = put here the path and the name of the original netCDF file  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants to get the average temperature for each season all in one new netCDF file.The input will be netCDF, the output will be netCDF(one file per model). example 2 is for several models; | import numpy as np import pandas as pd import xarray |
| seasonal\_mean\_3D\_sepM(file, save\_nerCDF='NO') | example1\_seasonal\_mean\_3D\_sepM.py example2\_seasonal\_mean\_3D\_sepM.py | This function will open an 3D netcDF file, compute the time mean for each season and save the new file in netCDF.  The date indicated for each season is the last time step of the corresponding season.  Each season will be saved in a separate file. | file = put here the path and the name of the original netCDF file  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save without mentioning ".nc"  (the program will add the seasons at the end of the name and the .nc)  if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants to get the average temperature for each season in four separate netCDF files (one per season).The input will be netCDF, the output will be netCDF(one file per model). example 2 is for several models; | import time import xarray import glob, os |
| seasonal\_mean\_1D\_sepM\_list(input, list, varName, save\_csv='NO') | example\_seasonal\_mean\_1D\_sepM\_list.py | This function will open a list with 1D netcDF files, compute the time mean for each season and save   the new files in 4 cvs files, one for each season; Each netcdf file will be saved on a different column in the csv file.  The date indicated for each year is the last time step of the corresponding season in the year. | input = put here the path to the 1D netCDF files  list = put here the list with the names of all 1D netCDF files you want  varName = put here the name of the variable, ex. 'tas'  save\_csv= if you want to save the file put here the path and the name of the csv file to save without .csv  (the program will add the seasons at the end of the name and the .csv)  if you don't want to save it, put 'NO' and use the file locally for other operations. | User wants to get the average temperature for each season in four separate CSV files specifically for one point (example Quebec City), for all models; each CSV file will containe all models; You will need to use another function before this function to select the point from the 3D netCDF files. The input will be several 1D netCDF files, the output will be 4 CSV file, one per season. | import time import xarray import glob, os import pandas as pd |
| annual\_smaller\_count\_3D(file, threshold, varName, show\_newPer='YES', save\_nerCDF='NO') | example2\_annual\_smaller\_count\_3D.py | This function will open an 3D netcDF file, for each year and for each grid point counts the number of values smaller than the threshold and save the new file in netCDF. The date indicated for each year is the last time step of the corresponding year. | file = put here the path and the name of the original netCDF file  threshold = value use as threshold  varName = name of variable  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | User asks for the number of days per year that received less than 6 mm of precipitation (can only be used with daily data)The input will be netCDF and the output will be netCDF. | import numpy as np import pandas as pd import xarray |
| annual\_greater\_count\_3D(file, threshold, varName, show\_newPer='YES', save\_nerCDF='NO') | example2\_annual\_greater\_count\_3D.py | This function will open an 3D netcDF file, for each year and for each grid point counts the number of values greater than the threshold and save the new file in netCDF. The date indicated for each year is the last time step of the corresponding year. | file = put here the path and the name of the original netCDF file  threshold = value use as threshold  varName = name of variable  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | User asks for the number of days per year that received more than 3 mm of precipitation (can only be used with daily data)The input will be netCDF and the output will be netCDF. | "import numpy as np import pandas as pd import xarray" |
| annual\_smallerEQ\_count\_3D(file, threshold, varName, show\_newPer='YES', save\_nerCDF='NO') | example2\_annual\_smallerEQ\_count\_3D.py | This function will open an 3D netcDF file, for each year and for each grid point counts the number of values smaller than or equal to the threshold and save the new file in netCDF. The date indicated for each year is the last time step of the corresponding year. | file = put here the path and the name of the original netCDF file  threshold = value use as threshold  varName = name of variable  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | User asks for the number of days per year that received less than or equal to 5 mm of precipitation (can only be used with daily data)The input will be netCDF and the output will be netCDF. | "import numpy as np import pandas as pd import xarray" |
| annual\_greaterEQ\_count\_3D(file, threshold, varName, show\_newPer='YES', save\_nerCDF='NO') | example2\_annual\_greaterEQ\_count\_3D.py | This function will open an 3D netcDF file, for each year and for each grid point counts the number of values greater than or equal to the threshold and save the new file in netCDF. The date indicated for each year is the last time step of the corresponding year. | file = put here the path and the name of the original netCDF file  threshold = value use as threshold  varName = name of variable  show\_newPer= put 'YES' if you want to verify the time dimention information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | User asks for the number of days per year that received more than or equal to 5 mm of precipitation (can only be used with daily data)The input will be netCDF and the output will be netCDF. | "import numpy as np import pandas as pd import xarray" |
| annual\_anomalies\_3D.py(file, first\_year, last\_year, show\_newPer='YES', save\_nerCDF='NO') | example1\_annual\_anomalies\_3D.py example2\_annual\_anomalies\_3D.py | This function will open an 3D netcDF file, compute the anomalies for each year based on the selected reference period and save the new file in netCDF. The date indicated for each year is the first time step of the corresponding year. | file = put here the path and the name of the original netCDF file first\_year= the first year of the desired reference period last\_year= the last year of the desired reference period  show\_newPer= put 'YES' if you want to verify the time dimension information  save\_nerCDF= if you want to save the file put here the path and the name of the netCDF file to save;   if you don't want to save it, put 'NO' and use the file locally for other operations. | User asks for the annual precipitation anomalies from a 1980-2005 reference period for Sept-Rivières—Caniapiscau, QC, Canada (latitude=55; longitude=-70). The input will be netCDF and the output will be netCDF. | import time import xarray import glob,os |
| test\_distribution.py(input, fld, output, var, x\_eval, legend, ylabel, xlabel) | example1\_test\_distribution.py example2\_test\_2distribution2.py | The script uses the kde function and has as objective to plot the distribution of values in a 1D netcdf file. The distribution is represented by an empirical PDF (Probability Distribution Functions) of values. | input= put in input the path to the folder with your netCDF files fld= put here the name of the netcdf file output= put in output the path to the folder where you want to save the figure with the name of the figure var= put here the name of the netCDF variable font\_size= place value for size of font in empirical pdf chart x\_eval= used to set the extent of the chart for variables used legend= insert title for variable represented by line ylabel= label for the y-axis of the chart xlabel= label for the x-axis of the chart (normally variable, include units) | User asks for comparison of statistical downscaling methods based on their distributions to assess which method is best to be used. The input will be netCDF and the output will be an empirical PDF. | import xarray import numpy as np import matplotlib.pyplot as plt from scipy import stats from scipy.stats import gaussian\_kde |
| CDF.py(input, fld1, fld2, output, var, x\_ticks, y\_ticks, legend1, legend2, ylabel, xlabel) | example\_CDF.py | The script has an objective to plot the distribution of values in a 1D netcdf file for precipitation. The distribution is represented by a Cumulative Distribution Functions (CDF). | input= put in input the path to the folder with your netCDF files fld1= put here the name of the netcdf file fld2= put here the name of the netcdf file to compare to output= put in the path to the folder where you want to save the figure with the name of the figure var= put here the name of the netCDF variable font\_size= set the size for font used in graph x\_ticks= value for tick interval on x axis y\_ticks= value for tick interval on y axis legend1= insert title for variable represented by line for fld1 legend2= insert title for variable represented by line for fld2 ylabel= label for the y-axis of the chart xlabel= label for the x-axis of the chart (normally variable, include units) | User asks for comparison of statistical downscaling methods based on their distributions to assess which method is best to be used for precipitation. The input will be netCDF and the output will be an empirical CDF. | import xarray import numpy as np import matplotlib.pyplot as plt from scipy import stats from scipy.stats import gaussian\_kde import statsmodels.api as sm |
| violinPlot.py(input, outputViolin, list\_nc, list\_names, palette, var, oy\_label, ox\_label, plot\_title, font\_size) | example\_violinPlot.py | This function will open specified netCDF files to create a probability density plot of values along with a box plot of the value range creating a violin plot as the end product to show multimodal distributions. | input= location folder for netCDFs outputViolin= location where the violin plot png image will be placed upon completion of script list\_nc= place here the list of netCDFs to be processed by the script list\_names= place the names of the files to compare as they are listed in list\_nc palette= used to define the colours used to represent the different datasets, the colours are quite numerous as supported with seaborn library var= place the field title for the variable to compare datasets oy\_label= label for the y-axis variable name with unit ox\_label= label for the different datasets being compared plot\_title= title for the violin plot font\_size= spcify size for font on plot | User asks for comparison of distribution for multimodal (multiple peak) netCDF data. Useful for comparing the different distribution of data to see where the variation in values occur. The input is netCDF files and the output is one violin plot png. | import numpy as np import pandas as pd import matplotlib.pyplot as plt import xarray as xr import seaborn as sns |
| TIMEmean\_3D.py | example2\_TIMEmean\_3D.py | This function will open an 3D netcDF file, compute the time mean over the entire period and save the new file in netCDF. | input= place the location of the folder containing netCDF files to be processed output= place the location where the new mean files will be list=glob.glob(' ') specify the file name to be processed | User asks for the mean of climate indices data over a selected time period. This function will calculate the mean for the entire dataset accessed so the desired time period must be defined before running this script. The script takes netCDF files as input and will produce new netCDF files with function name appended onto the files' end. | import time import numpy as np import pandas as pd import xarray import matplotlib.pyplot as plt import glob, os |