TMTplus Introduction to Scientific Programming

Mahdi Farnaghi, Mahdi Khodadadzadeh & Robert Ohuru

March 2021

Note from the teaching staff

The materials that we use for this course are somewhat new and have been heavily reworked. We will likely have made mistakes in this work or have glossed over issues that deserved better or different treatment. Where you find issues worth noting please do report these to us as it will allow to repair and improve.

By the way, welcome to the Exercise book for Introduction to Scientific Programming! We believe this book is self-explanatory, so go ahead and practice.

Chapter 10

Gdal raster operations

In many geospatial projects, image handling functionality is fundamental to achieve success.

The exercises

The goal of this practical session is to learn using the *gdal* package and handle raster data. These exercises come in two parts: the first part in which you will get in touch with the *gdal* Python library to work with raster datasets and the second part were you should use the *gdal* executables and scripts. Naturally, these exercises cannot be used in isolation as they relate to you have previously learned.

Our intention is that you spend most of your time with the first exercises, and use this session a starting point in the use of *gdal* in Python. Thus, we keep following an incremental approach of the exercises, by gradually increasing its complexity.

11.1 *Gdal* exercises

In this section, you will learn the basic handling of raster data with *gdal*. It is not possible to cover all the functionalities and functions available in this package, thus, we encourage you to keep checking the *gdal* documentation in the website to find out how to properly use them.

- *gdal* documentation

- *gdal* format list
- *gdal* tutorials
- *gdal* utilities

Ex 10.1 Open the file `2014.tif` that represents the daily maximum temperature of 2014, and obtain the following pieces of information about the file:

1.1 Driver name

1.2 Raster size

1.3 Geo transform information:

1.3.1 Top left coordinates

1.3.2 Pixel size

1.3.3 Rotation

1.4 Number of bands

Ex 10.2 Within the dataset `2014.tif`, access the first day of measurements (i.e., the first band) and obtain the statistics of those measurements. Also to better understand this data, determine the no data value (if defined) and check the number of overviews.

Ex 10.3 Within the dataset `2014.tif`, access the first day of measurements again, and determine the temperature for position $(x, y) = (200, 137)$.

Ex 10.4 With the same dataset and that first day, extract a subset for the Amsterdam area, and save it as a raster file named `2014_amsterdam.tif` using the GTiff driver.

Data for the Amsterdam area can be extracted using the function

`gdarr.BandReadAsArray(band, xoff, yoff, win_xsize, win_ysize)`

with the following parameter values: `xoff= 0`, `yoff=0`, `win_xsize=200`, `win_ysize=200`. Verify whether these values make sense to you.

Ex 10.5 Transform dataset `2014.tif` into a numpy array, then next determine temperature values for the following day/location combinations.

You may use function `gdarr.DatasetReadAsArray(dataset, xoff, yoff, win_xsize, win_ysize)` for the transformation . Do not forget that in numpy X represents columns and Y represents rows.

Days	Pixel position	Temperature
1	0, 0	
31	100, 100	
59	78, 150	
90	186, 180	
120	20, 160	
151	100, 100	
365	78, 150	

With the same dataset and the first day measurements, extract a subset of Enschede area and save it into a raster file named 2 014_enschede.tif using the GTiff driver. **Ex 10.6**

The Enschede area can be extracted using the function using parameters: xoff= 100, yoff=40, win_xsize=200, win_ysize=200. Hint: you will need a new geotransform using pixel offset calculations. Check the slides how to do this.

11.2 The use of .Translate() and .Warp() methods

In this block of exercises, you learn how to run the gdal.Translate and gdal.Warp methods.

You should have a folder called 2014_csv. This folder has 31 csv files, one per day, each of which represents the maximum temperature in the month of January 2014. With the method gdal.Translate you should create new raster tif files based on the csv files. The coordinate system of the csv files is EPSG code 28992. Ensure that you specify the nodata value as -9999. **Ex 10.7**

You will need to create a loop using the os.listdir(directory) function. For example, you can use the following code to loop over files in a directory:

```
for filename in os.listdir(directory):
    if filename.endswith(".tif"):
        print(filename)
```

Check the file size of the tif files; do you see any change compared to those of the csv files? Load at least one tif file into a GIS software of your choice to visualize the image file, and build your intuition.

Try to solve Exercise 4 and 6 above with gdal.Translate(). **Ex 10.8**

In Exercise 7, we should have created 31 tif files, one per day. These files make use of the spatial reference system known as EPSG code 28992. Now, reproject these files into new tif files, that make use of spatial reference system known as EPSG code 4326 (WGS84), through the use of gdal.Warp(). Please check the presentation slides for more information on how to do this. **Ex 10.9**

And do also check Warp options for the list of available gdal.Warp options.

Congratulations! You have finished your first raster data analysis in Python.