How to Create a Reproducible Workflow for Downloading and Processing Gridded Weather Data

Lucas Graunke, TMDL Writer

June 22nd, 2023

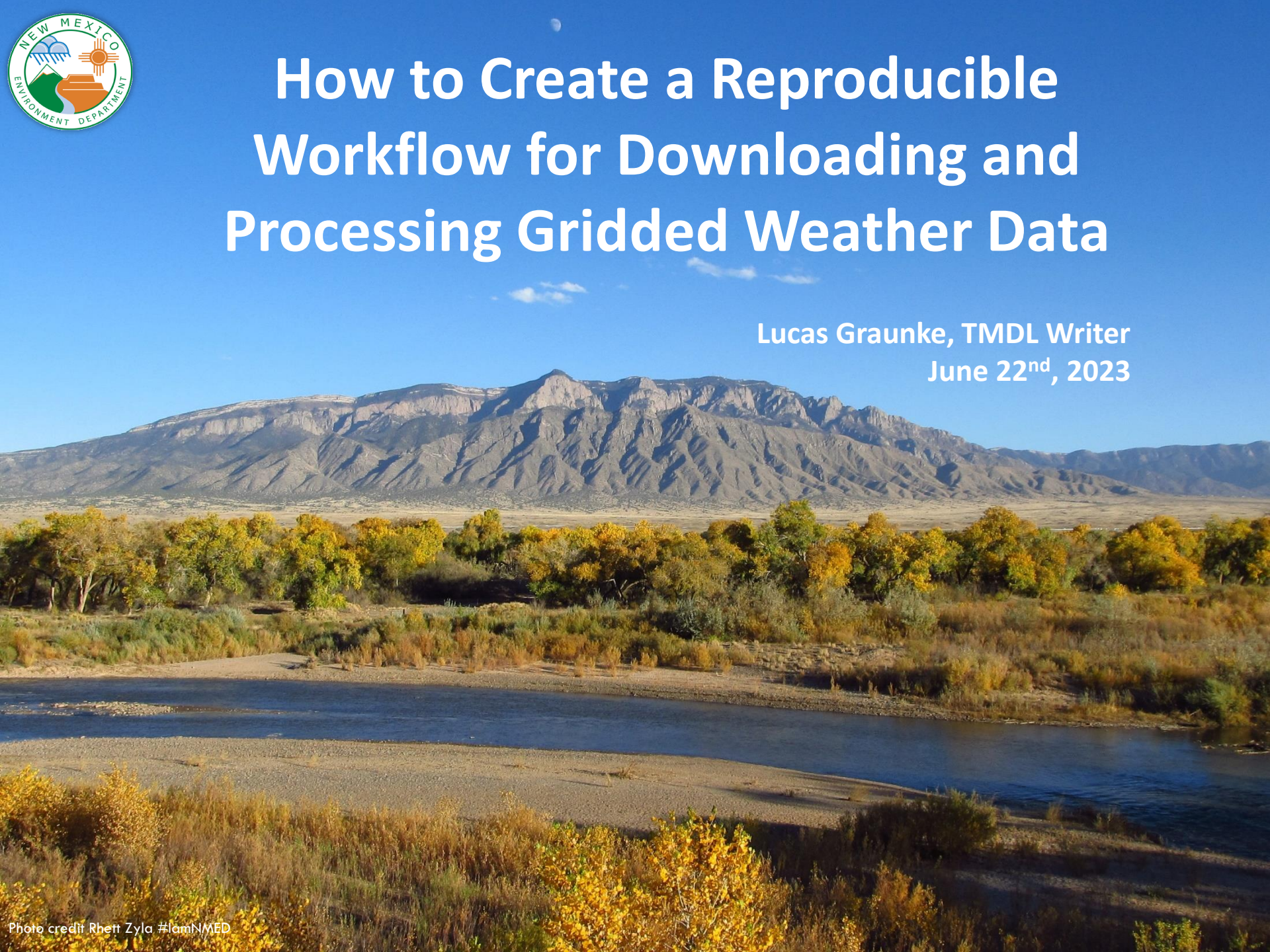


Photo credit Rhett Zyla #IamNMED

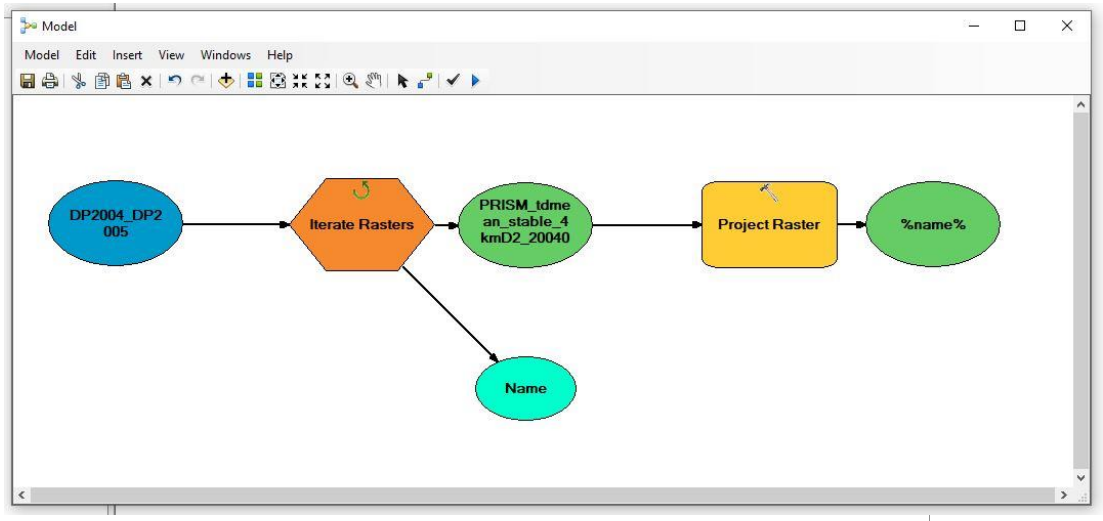
Why create gridded weather data?

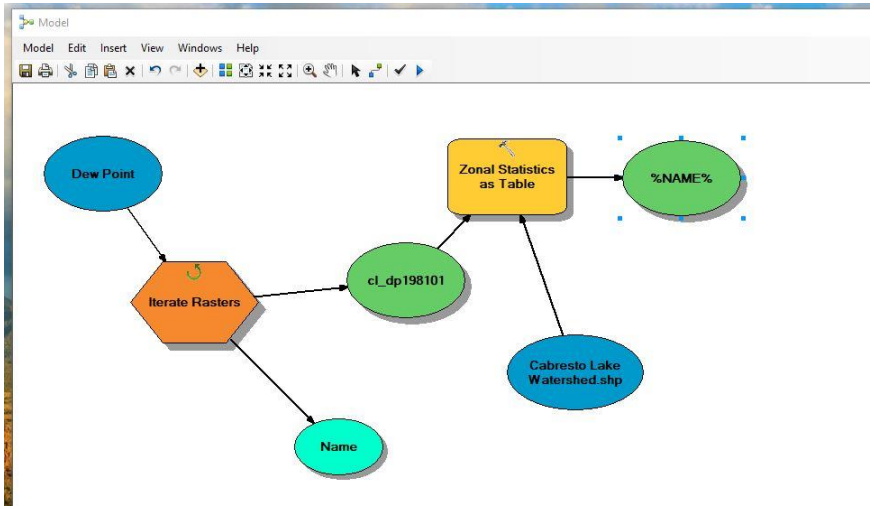
- Gridded weather data is important for modeling input, especially in states with few climate stations
- New Mexico has very limited long term climate stations for a variety of stations
- The watersheds being modeled are extremely remote

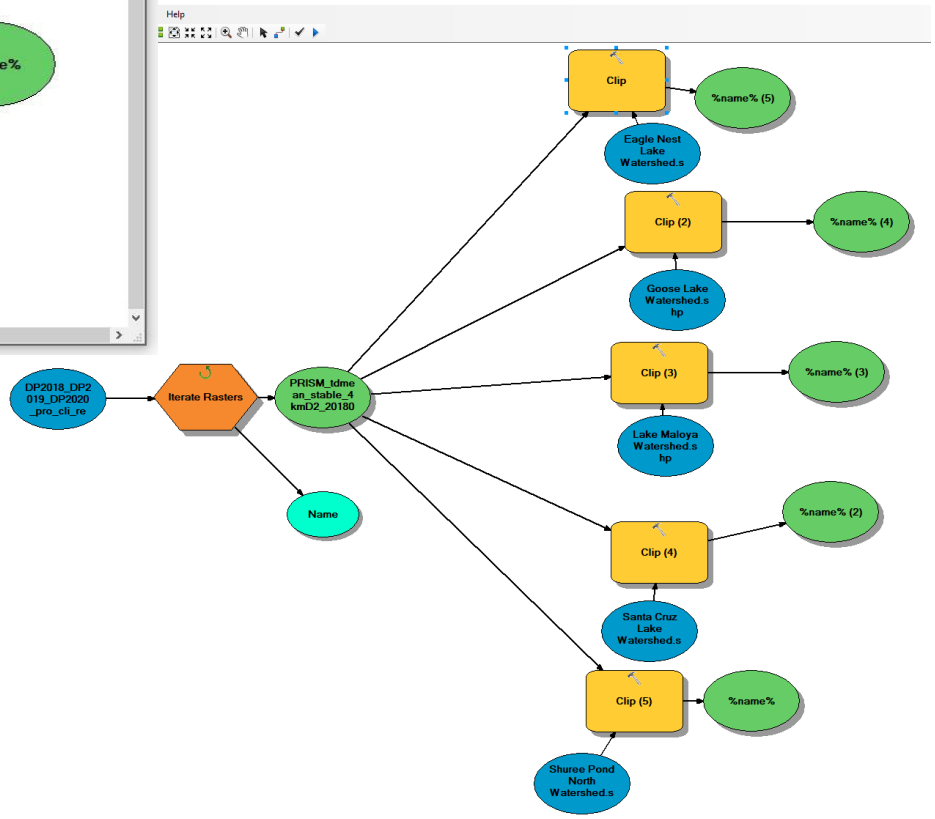
Steps for creating gridded weather data

1. Download climate rasters from a database like PRISM
2. Reproject climate rasters to correct projection
3. Clip climate rasters to watersheds or catchments
4. Use ArcGIS/Python/R to run zonal statistics on the reprojected and clipped climate rasters

Brute Force and Model Builder







Downloading Climate Rasters

- A Jupyter Notebook was used to write out the Python script needed to bulk download and unzip the PRISM data.
- Need to create a working directory, choose a resolution for the rasters, and decide what dates/time period to download.

How to Bulk Download PRISM Data

These are the packages needed to complete the tasks below.

In [14]:
```
import os, shutil, wget
import urllib.request
```

Create a working directory where the downloaded PRISM data will be stored. The second line of code here will create a working directory if the one specified does not exist.

In [15]:
```
working_directory = r'C:\Users\lucas.graunke\Desktop\Lakes\PRISM Data\EPA Code\Practice Downloading'
if not os.path.exists(working_directory): os.makedirs(working_directory)
```

This code is what actually bulk downloads the data. With the help of a FOR loop, many multiple rasters can be downloaded at once. Specify they years, months and days that need to be downloaded. If only monthly data is needed, delete the "days" line, and if only annual data is needed then delete both the "months" and "days" lines.

In [16]:
```
years  = ('1995', '1996')
months = ('01', '02',)
days   = ('01', '02', '03', '04', '05')
for year in years:
    for month in months:
        for day in days:

            url = 'http://services.nacse.org/prism/data/public/4km/ppt/' + year + month + day
            savedFile = savedFile = os.path.join(working_directory, 'PRISM_4km_ppt_' + year + month + day + '.zip')

            wget.download(url, savedFile)
```

This download data will need to be unzipped. The code below will unzip all zipped files in the working directory.

In [ ]:
```
from zipfile import ZipFile

zip_file_location = os.path.join(working_directory, 'PRISM_4km_ppt_19950101.zip')

with ZipFile(zip_file_location, 'r') as zipObj:
   # Extract all the contents of zip file in current directory
   zipObj.extractall(working_directory)

for file in os.listdir(working_directory):
    if file.endswith('.zip'):
        filePath = os.path.join(working_directory, file)
        with ZipFile(filePath, 'r') as zipObj:
            zipObj.extractall(working_directory)
```

Reprojecting Climate Rasters

- The rasterio package was used reproject the climate rasters.
- EPSG codes were used to define the new project. Information on EPSG codes can be found here: https://spatialreference.org/

How to Project Climate Rasters

In [18]:
```
import rasterio, os
from rasterio.warp import calculate_default_transform, reproject, Resampling
```

This is defining the projection that we want the rasters to be in.

In [25]:
```
dstCrs = {'init': 'EPSG:4326'}
```

This is defining the source path and pulling the list of rasters in that directory.

In [26]:
```
srcPath = 'C:/Users/lucas.graunke/Desktop/GIS_Data/PRISM Data/'

lstRst = [r for r in os.listdir(srcPath) if r.endswith('.bil')]
lstRst
```

Out[26]:
```
['PRISM_tdmean_stable_4kmM3_1981_bil.bil',
 'PRISM_tdmean_stable_4kmM3_1982_bil.bil',
 'PRISM_tdmean_stable_4kmM3_1983_bil.bil']
```

This prints the current projection of the rasters.

In [27]:
```
expRst = rasterio.open(srcPath+lstRst[0])
expRst.crs
```

Out[27]:
```
CRS.from_wkt('GEOGCS["NAD83",DATUM["North_American_Datum_1983",SPHEROID["GRS 1980",6378137,298.257222101,AUTHORITY["EPSG","7019"]],AUTHORITY["EPSG","6269"]],PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199433],AXIS["Longitude",EAST],AXIS["Latitude",NORTH]]')
```

This defines the folder that the projected rasters will be placed in.

In [28]:
```
dstPath = 'C:/Users/lucas.graunke/Desktop/GIS_Data/PRISM Data/Projected/'
```

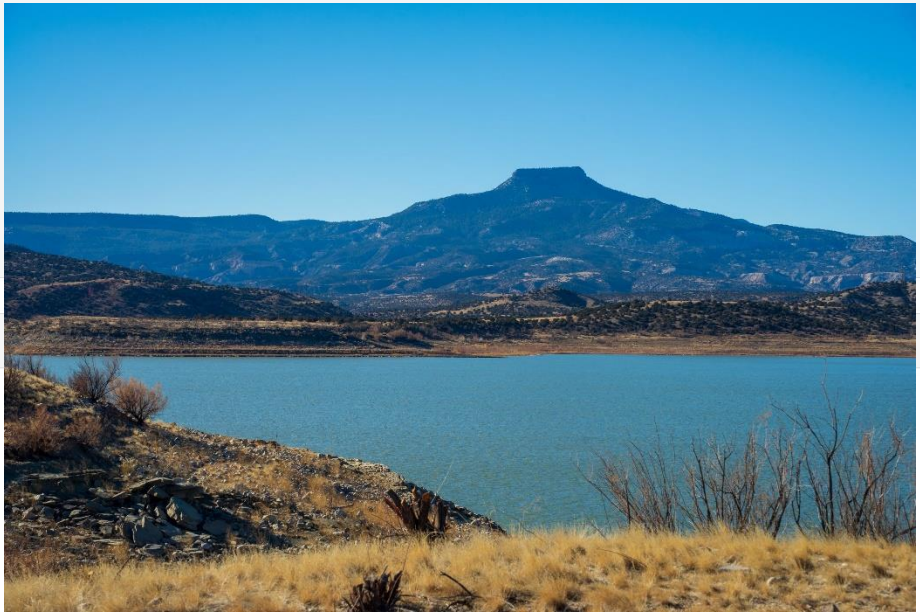
This is pulling everything together to define how to reproject the data. The FOR loop below will reprojects all of the rasters located within the source path defined above.

In [29]:
```
def reprojectRaster(srcRst, dstRst, dstCrs, srcPath, dstPath):
    srcRst = rasterio.open(srcPath+srcRst)

    transform, width, height = calculate_default_transform(
        srcRst.crs, dstCrs, srcRst.width, srcRst.height, *srcRst.bounds)

    kwargs = srcRst.meta.copy()
    kwargs.update({
            'crs': dstCrs,
            'transform': transform,
            'width': width,
            'height': height
        })


    dstRst = rasterio.open(dstPath+dstRst, 'w', **kwargs)

    for i in range(1, srcRst.count + 1):
        reproject(
            source=rasterio.band(srcRst, i),
            destination=rasterio.band(dstRst, i),
            src_crs=srcRst.crs,
            dst_crs=dstCrs,
            resampling=Resampling.nearest)

    dstRst.close()
```

This is the code that acutally reprojects the data.

In [30]:
```
for srcRst in lstRst:
    dstRst = srcRst[:-4]+'_reprojected'+srcRst[-4:]
    reprojectRaster(srcRst,dstRst,dstCrs,srcPath,dstPath)
    print('Reprojection of %s done'%srcRst)
```

```
Reprojection of PRISM_tdmean_stable_4kmM3_1981_bil.bil done
Reprojection of PRISM_tdmean_stable_4kmM3_1982_bil.bil done
Reprojection of PRISM_tdmean_stable_4kmM3_1983_bil.bil done
```



Clipping Climate Rasters

- The gdal function was used to batch clip the rasters here.
- More information on gdal can be found here: https://gdal.org/index.html

How to Batch Clip Rasters

In [ ]:
```
import os, fnmatch
from osgeo import gdal
```

In [ ]:
```
inFolder = "C:/Users/lucas.graunke/Desktop/GIS_Data/PRISM Data/Projected/"
outFolder = "C:/Users/lucas.graunke/Desktop/GIS_Data/PRISM Data/Projected/Clipped/"
mask = "C:/Users/lucas.graunke/Desktop/GIS_Data/Random/new_mexico_bnd/new_mexico_bnd.shp"
os.chdir (inFolder)
```

In [ ]:
```
def findRasters (path, filter):
    for root, dirs, files in os.walk(path, filter):
        for file in fnmatch.filter(files, filter):
            yield os.path.join (root, file)
```

In [ ]:
```
for raster in findRasters (inFolder, '*.bil'):
    (infilepath, infilename)= os.path.split (raster)
    print(infilename)
    outRaster= outFolder+ 'clip_'+ infilename
    print(outRaster)
    warp= gdal.Warp(outRaster,raster, cutlineDSName = mask, dstNodata = 255)
```

Using rasterstats on Climate Rasters

- Spyder was used to write this script because of its similarity to R Studio.
- The rasterstats function is used here to find the mean, max or min of the climate rasters called upon. This script is finding the mean daily dew point for each of the daily rasters for Eagle Nest Lake.

```
##########################################################################################################

##########################################################################################################

# This step creates an empty pandas DataFrame. This is the dataframe that will be populated with     #
# the processed climate data.                                                                         #

ENL_AvgDP = pd.DataFrame('', columns = ['Count', "Average_DP_Eagle"], index = np.arange(1, 10000))

i = 1

##########################################################################################################

##########################################################################################################

# This step creates a for loop that processes the information from TMax rasters. Rasterstats is      #
# used to get the zonal stats from the TMax rasters.                                                  #

for rast in os.listdir(r'D:/Desktop/GIS Data/Daily Data/RasterStats/Eagle Nest Lake/Dew Point'):
   if rast[-4: ] == '.bil':

        dpavg = rasterio.open(r'D:/Desktop/GIS Data/Daily Data/RasterStats/Eagle Nest Lake/Dew Point' + '\\' + rast)
        dpavg_array = dpavg.read(1)
        affine = dpavg.transform
        nodata = -9999

        # Using rasterstats to obtain zonal stats for Shuree Pond North
        average_dp_eagle= rasterstats.zonal_stats(watershed_polygon,
                                                  dpavg_array,
                                                  affine = affine,
                                                  nodata = nodata,
                                                  stats = ['mean'],
                                                  geojason_out = True)

        average_dp_eagle = average_dp_eagle[0]['mean']

        ENL_AvgDP.loc[i]['Count'] = rast
        ENL_AvgDP.loc[i]['Average_DP_Eagle'] = average_dp_eagle
        print(rast)
        print(average_dp_eagle)

        i = i + 1

##########################################################################################################

##########################################################################################################
```

ENL_AvgDP - DataFrame

Index	Count	Average_DP_Eagle
1	PRISM_tdmean_stable_4kmD2_19950101_bil.bil	-12.8628
2	PRISM_tdmean_stable_4kmD2_19950102_bil.bil	-16.6303
3	PRISM_tdmean_stable_4kmD2_19950103_bil.bil	-13.1589
4	PRISM_tdmean_stable_4kmD2_19950104_bil.bil	-11.1799
5	PRISM_tdmean_stable_4kmD2_19950105_bil.bil	-8.55681
6	PRISM_tdmean_stable_4kmD2_19950106_bil.bil	-5.98687
7	PRISM_tdmean_stable_4kmD2_19950107_bil.bil	-10.5767
8	PRISM_tdmean_stable_4kmD2_19950108_bil.bil	-8.01498
9	PRISM_tdmean_stable_4kmD2_19950109_bil.bil	-7.21357
10	PRISM_tdmean_stable_4kmD2_19950110_bil.bil	-7.45817
11	PRISM_tdmean_stable_4kmD2_19950111_bil.bil	-9.09032
12	PRISM_tdmean_stable_4kmD2_19950112_bil.bil	-6.65923
13	PRISM_tdmean_stable_4kmD2_19950113_bil.bil	-7.60044
14	PRISM_tdmean_stable_4kmD2_19950114_bil.bil	-9.31272
15	PRISM_tdmean_stable_4kmD2_19950115_bil.bil	-7.35618
16	PRISM_tdmean_stable_4kmD2_19950116_bil.bil	-7.2872

Useful Links

- PRISM Website: https://prism.oregonstate.edu/
- EPA Modeling Webinar for Processing Gridded Weather Data: https://www.youtube.com/watch?v=-QtNOWs1FSM
- Github Directory with scripts: https://github.com/KateriSalk/EPA_WaterQualityModeling_Webinars
- Rasterstats Python info: https://pythonhosted.org/rasterstats/

Contact Information:

Email: lucas.graunke@env.nm.gov
Phone: (505) 819-9823