Understanding Meteosat satellite data and converting said data into useful products such as NDVI and FOG

Syed Mustafa Haider

Gis Intern

National Aerospace Science and Technology Park (NASTP)

12/1/2023

Table of Contents

**[List Of Figures](#_Toc152586561)** [3](#_Toc152586561)

[**1.** **Introduction** 4](#_Toc152586562)

[**1.2 MSG Instrument GERB** 4](#_Toc152586563)

[**1.3 MSG Instrument SEVIRI** 4](#_Toc152586564)

[1.4 MSG Products 5](#_Toc152586565)

[**2. Downloading data** 7](#_Toc152586566)

[**3. Managing the python libraries for this project** 8](#_Toc152586567)

[**3.1 Getting started with Anaconda** 12](#_Toc152586568)

[**3.1.1 Installing Packages through Anaconda Navigator.** 13](#_Toc152586569)

[**3.1.2 Installing Packages through Anaconda Prompt.** 14](#_Toc152586570)

[**3.2 Getting started with Satpy** 15](#_Toc152586571)

[**3.2.1 Creating NDVI from HRIT data Of MSG** 15](#_Toc152586572)

[**3.2.2 Creating Fog Composite from HRIT data Of MSG** 19](#_Toc152586573)

# **List Of Figures**

[Figure 1: SEVIRI Channels 4](#_Toc152586381)

[Figure 2: Anaconda download 8](#_Toc152586382)

[Figure 3: Anaconda cloud 8](#_Toc152586383)

[Figure 4: Anaconda Installation Path 9](#_Toc152586384)

[Figure 5: Anaconda Installation Environment Path 10](#_Toc152586385)

[Figure 6: Anaconda Installation Complete 10](#_Toc152586386)

[Figure 7: Anaconda Navigator 11](#_Toc152586387)

[Figure 8: Creating Environments 1 11](#_Toc152586388)

[Figure 9: Creating Environment 2 12](#_Toc152586389)

[Figure 10: Installing Packages through anaconda navigator 12](#_Toc152586390)

[Figure 11: Starting Anaconda Prompt 13](#_Toc152586391)

[Figure 12: Anaconda Prompt 13](#_Toc152586392)

[Figure 13: HRIT data files 15](#_Toc152586393)

[Figure 14: Importing the necessary libraries 15](#_Toc152586394)

[Figure 15: Creating the scene 16](#_Toc152586395)

[Figure 16: NDVI Calculation 16](#_Toc152586396)

[Figure 17: NDVI result through matplotlib 17](#_Toc152586397)

[Figure 18:NDVI as geotiff 17](#_Toc152586398)

[Figure 19: NDVI as geotiff in ArcMap 18](#_Toc152586399)

[Figure 20: Importing libraries for fog 18](#_Toc152586400)

[Figure 21: Creating the Scene for Fog 19](#_Toc152586401)

[Figure 22: Creating the Fog composite 19](#_Toc152586402)

[Figure 23: The Fog Composite 20](#_Toc152586403)

[Figure 24: Exporting the scene as a tiff 20](#_Toc152586404)

# **Introduction**

The Meteosat satellites are operated as a two-satellite system providing detailed **full disc imagery** over Europe and Africa every 15 minutes and **rapid scan imagery** over Europe, every five minutes. Meteosat imagery is crucial for now casting, which is about detecting rapidly developing high impact weather and predicting its evolution a few hours ahead, in support of the safety of life and property. Meteosat satellites have been providing crucial data for weather forecasting since 1977. MSG (Meteosat Second Generation) consists of a series of four geostationary meteorological satellites, along with ground-based infrastructure, that will operate consecutively until 2020.The launch dates of Meteosat were:

* MSG-1/Meteosat-8: 28 August 2002
* MSG-2/Meteosat-9: 21 December 2005
* MSG-3/Meteosat-10: 5 July 2012
* MSG-4: Meteosat-11 15 July 2015

The MSG has an image repeat cycle of 15 min, with 30 min for (Meteosat) having 12 spectral channels. MSG consists of 4 satellites MSG 1, 2, 3 and 4. The primary instruments in the MSG series are the SEVIRI (Spinning Enhanced Visible and Infrared Imager) sensor and the GERB (Geostationary Earth Radiation Budget) instrument. GERB monitors the radiation budget in terms of climate change, food production and natural disaster prediction which is achieved by measuring the top layer of the atmosphere, the shortwave and emitted longwave of the region. GERB includes two instruments the IOU (Instrument Optical Unit) imager device and the Instrument Electronics Unit (IEU) data handling device.

## **1.2 MSG Instrument GERB**

The primary use or objective of GERB is to observe the Earth’s radiation budget. The GERB instrument is on MSG 1 and MSG 2. GERB monitors the radiation budget in terms of climate change, food production and natural disaster prediction which is achieved by measuring the top layer of the atmosphere, the short-wave and emitted longwave of the region. GERB includes two instruments the IOU (Instrument Optical Unit) imager device and the Instrument Electronics Unit (IEU) data handling device

## **1.3 MSG Instrument SEVIRI**

SEVIRI is an abbreviation for Spinning Enhanced Visible and Infrared Imager which in simpler terms has 12-channel imager observing the earth-atmosphere system, from which 11 channels observe the full disk of the earth with a 15-min repeat cycle. The HRV (High resolution Visible) channel covers half of the full earth disk in the east to west direction and full disk in the north-south direction. The HRV has a spatial resolution of 1.67 km. The detail of the various channels are shown below:

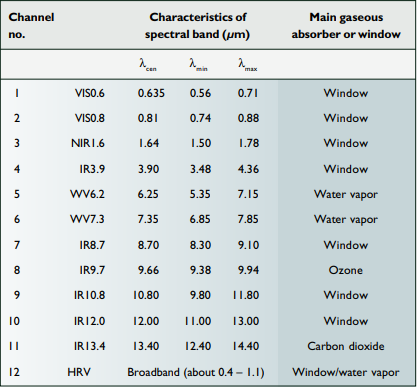


Figure 1: SEVIRI Channels

## 1.4 MSG Products

There are various products produced by the Meteosat series, the operational meteosat meteorological products are handled by the MPEF, the MTP (Meteosat transition program) products are specifically produced by the MPEF ( Meterological Product Extraction Facility) the core products are wind products extracted from three channels in near-real time. The current operational products are as follows:

**CLA (Cloud analysis products):** In simpler terms this product provides scene analysis results on a scale of 100 km x 100 km or better along with providing information on cloud cover, cloud-top temperature, cloud type and phase and assisting the generation of atmospheric motion vectors (AMV) which can provide information on wind speed and direction.

**Cloud-top height (CTH):** as the name implies this product is a derived product image which provides information regarding cloud height at a resolution of 3 x 3 super pixel resolution. The main use of this product is in aviation meteorology as it provides hights with a vertical resolution of 300m.

**Clear-Sky radiance:** This product provides the mean radiance of cloud free pixels in [W m−2 sr−1 (cm−1) −1]

**Tropospheric humidity (TH):** This product provides estimates of mean-relative humidity for two tropospheric layers. The first layer humidity product is referred to as upper tropospheric.

**Atmospheric motion vectors:** These products in simpler terms are very important as far as numerical whether prediction. This product would be derived from cloud and water vapor motion using primarily the 0.6- or 0.8-µm channel, the 10.8-µm channel, and the 6.2- and 7.3-µm channels, respectively

**Climate dataset (CDS):** provides statistical information about the scene classes in a processed segment 984 | JULY 2002 (nominally 32 × 32 pixels). It is a concise summary of the radiances observed in a segment and potentially very useful for climatological studies of cloud and radiation fields.

**Total ozone product (TOZ):** This uses the 9.7-µm channel and other SEVIRI channels and correlative data, and is derived with a regression algorithm (Orsolini and Karcher 2000; Engelen et al. 2001). The ozone observations are useful for monitoring and forecasting UV radiation at the ground level.

Then there are SAF products which are satellite application facilities some of he examples are as follows:

* **Examples of products from the Ocean and Sea Ice SAF include**

1. Atlantic sea surface temperature,
2. surface radiative fluxes over the Atlantic,
3. sea ice (polar Atlantic), that is, ice edge/cover/type.

• Examples of products from the Climate SAF are

1. cloud parameters,
2. surface radiation budget components,
3. radiation budget components at the top of the atmosphere,
4. Humidity composite products.

• Examples of targeted products from the Land Surface Analysis SAF are

1. vegetation parameters and biophysical indicators,
2. snow cover,
3. land surface temperature,
4. soil moisture,
5. surface albedo,
6. evapotranspiration, and
7. Shortwave and longwave radiation parameters.

Other primary features that can be extracted from the MSG data are as follows:

**Visable imagery:** As the name implies it refers to satellite imagery in the visable spectrum in primarily daylight conditions which can be used for monitoring cloud cover, land cover and atmospheric conditions.

**Infrared Imagery:** As the name implies this referes to the imagery in the infrared specterumn which primarily deals with capturing temperature variations in the earth’s surface and atmosphere. This can be used from monitoring cloud-top temperatures, identifying areas of precipitation and temperature data.

**Water Vapor imagery:** this corresponds to the collection of atmospheric moisture patterns and humidity variations.

**Multispectral Imagery:** refers to the creation of composite images that can be used in land-use classification and vegetation monitoring.

**Derived Products:** As the name implies these are products derived from satellite data such as sea surface temperatures, atmospheric motion vectors and etc. These products can be used in weather forecasting, climate monitoring and etc.

These datasets are often provided in standard formats such as NetCDF, HDF, or GeoTIFF. The specific format may depend on the data distribution policies of EUMETSAT or the organization providing access to the satellite data.

# **2. Downloading data**

This section deals with the basic techniques and sources used to download the meteosat data. The data of any meteosat product can be downloaded from the website (<https://www.eumetsat.int/our-satellites/meteosat-series>) in order to access the data, registration to this afro mentioned website is required where like any other service providing website a person needs to provide his email and other information for registration purposes. Once registered there are various tools used to either visualize the data set or to download said dataset. The following table would represent the basic tools provided by EUMETSAT

Table 1: EUMETSAT tools

|  |  |  |
| --- | --- | --- |
| **Tool or feature provided by EUMETSAT** | **Link to said FEATURES and tools** | **Working** |
| EUMETSAT-View | https://view.eumetsat.int/productviewer?v=default | TO View the different meteosat products such as fog, air temperature, thunderstorm, etc. On an open web geoportal which allows a user to also download tiles in various formats. It assists in viewing various products with respect to the user’s requirement and also provides features such as WMS, WFS and WCS through the use of API keys. |
| EUMETSAT-Data Store | https://data.eumetsat.int/search?query= | AS the name implies this is the catalogue of EUMETSAT meteosat products, this tool allows users to access near- real time data sets of various products through and online GUI or through API. Data products chosen can also be queried for user specific demands through the layer editing tool which is enabled once the data chosen is added to the cart where their would be the option to “Customie Cart” where users can clip said data, choose specific bands from data and change the output format |
| EUMETSAT- Ordering Client/ Data Store | https://www.eumetsat.int/ordering-client | As the name implies this is the ordering client tool which allows user to access all the data provided by EUMETSAT and orders can be placed for user specific data sets |

Once the data has been downloaded the next step is to work on said data to produce specific products such as NDVI and FOG. For this purpose various tools such as ArcMap or Qgis can be used but for the case of this project python was used. The next session deals with the installation of python libraries and environment for production.

# **3. Managing the python libraries for this project**

The libraries necessary for the handeling of meteosat satellite data are as follows:

* Satpy
* Gdal
* Geoviews
* Folium
* Cartopy
* Netcdf4
* Geopandas

Table 2: Libraries

|  |  |
| --- | --- |
| **Library Name** | **Usage** |
| Satpy | Satpy is a python library designed for working with metrological satellite data along with remote sensing satellite data. It simplifies the process of reading, Processing and visualizing of satellite data by providing a unified and user friendly interface |
| Gdal | GDAL is a powerful open source python library for reading and writing ( data manipulation) raster and vector geospatial data. The GDAL python library often used in conjunction with the OGR (which handles vector data ) library provides a comprehensive set of tools for working with geospatial data formats |
| Geoviews | Geoviews is a python library designed for creating interactive visualizations of geospatial data. Geoviews simplifies the process of exploring and visualizing geographical information. |
| Cartopy | Cartopy is a python library designed for cartographic projections and geospatial data visualization and it provides tools for creating maps and etc. |
| Netcdf4 | This library is used for working with Netcdf files. Netcdf is a file format for storing multi-dimensional data such as climate, weather oceanography and atmospheric data |
| Geopandas | This library works with geospatial data by extending the capabilities of pandas library, basically it is a spatial extension of the pandas library. It provides data structure and functions to efficiently handle and analyze geospatial data sets. |
| Folium | It is a python library used for creating interactive and customizable maps (web maps) it is built on top of the leaflet JavaScript library and provides a simple and continent way to visualize spatial data |

Now these libraries of python can be installed through “pip installation” or simple cmd python installation but it would be better to install a python distribution handler that manages said libraries in its specific environment. One such distribution handler is Anaconda. In simpler terminology Anaconda is a package handler for both python and r programming language for the purpose of using said programming libraries for data science, machine learning and scientific computing. The primary purpose of Anaconda is to simplify the process of setting up and managing different packages (libraries) in an environment as it provides a comprehensive package management system. In order to install Anaconda a user registration is required and after registration the installation file is present on the website (<https://www.anaconda.com/download>) which is shown as bellow:

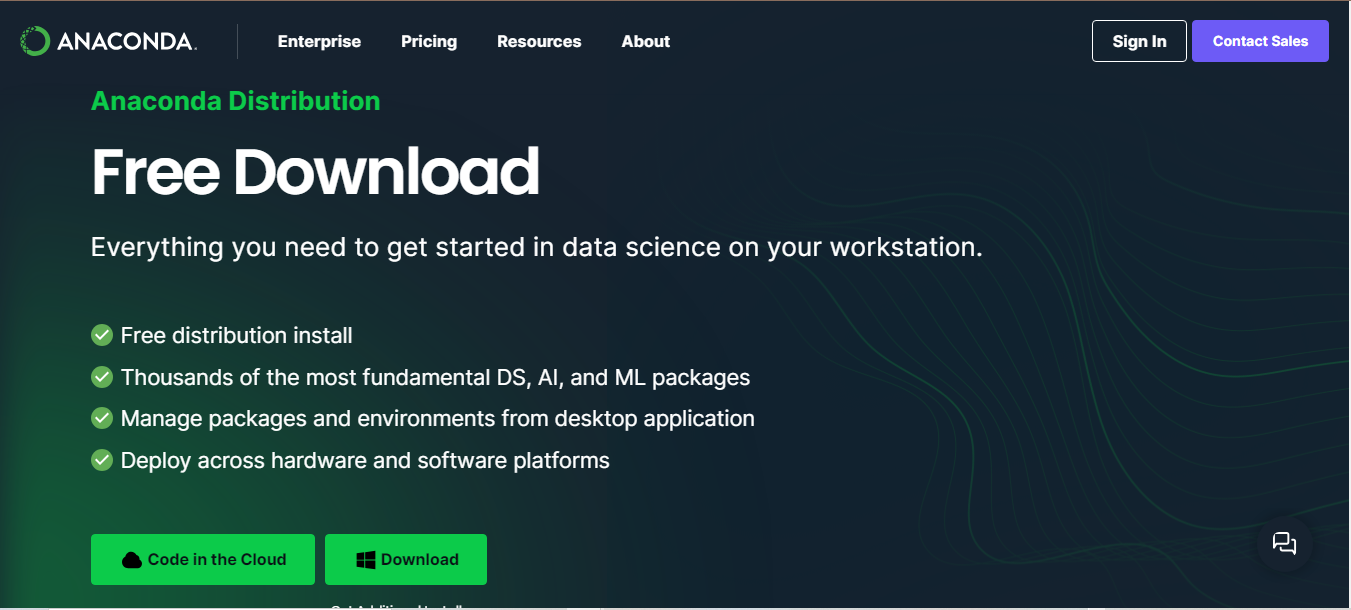


Figure 2: Anaconda download

While the user page after registration can be accessed from the following website (<https://anaconda.cloud>) is shown below:

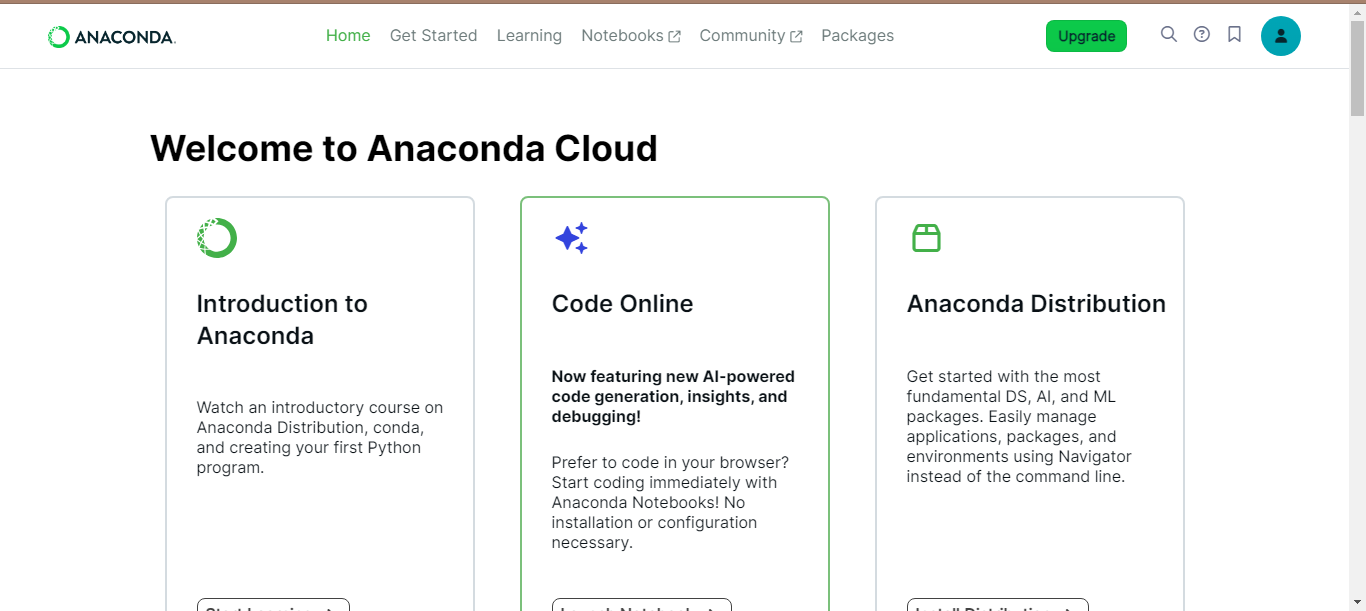


Figure 3: Anaconda cloud

This user page includes all the relevant documentation regarding different libraries used in anaconda for various purposes. After the downloading of the .exe file the next step is to install Anaconda on the OS of the user, the steps for installation are provided in the installation wizard GUI. The very first step in the installation is to choose the path to download anaconda in as follows:

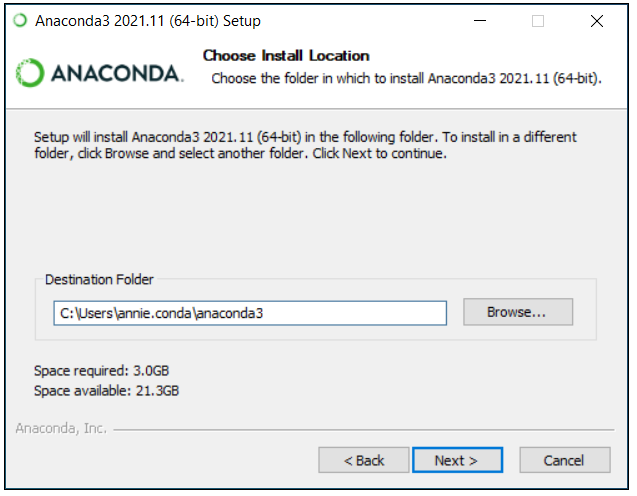


Figure 4: Anaconda Installation Path

After defining the path the **next step is very important** as the installation wizard GUI asks the user to choose **the PATH Environment**. The figure of the next step is as follows

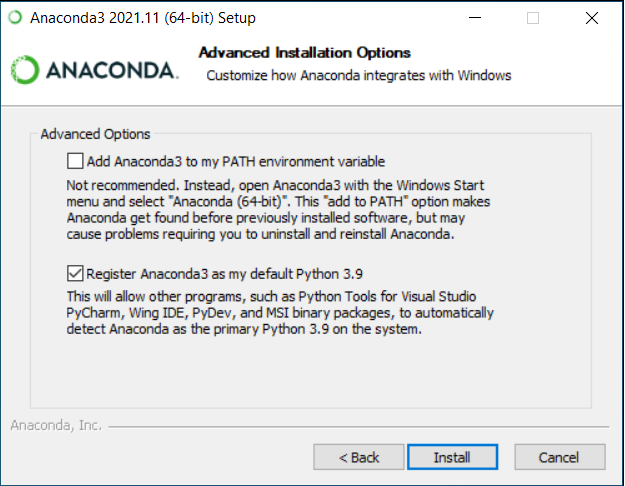


Figure 5: Anaconda Installation Environment Path

For this step it is important to choose the recommended option as the recommended option allows other coding interpreters such as visual studio and etc. to automatically detect anaconda as the default python environment. The first option might cause environmental issues. This step can also be avoider or rather it can be set to the recommended action by starting the **.exe file as administrator. After the afro mentioned step the installation would start and it would take a while before anaconda is fully installed in the users Operating System.** The Final popup would be as follows:

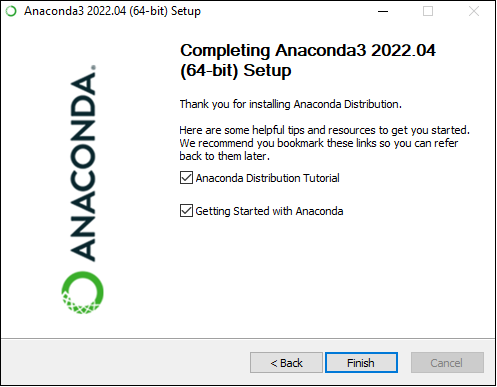


Figure 6: Anaconda Installation Complete

## **3.1 Getting started with Anaconda**

After the completion of the installation of anaconda in the user’s operating system the default desktop GUI of anaconda is called “Anaconda Navigator”. The following figure represents the starting GUI of anaconda

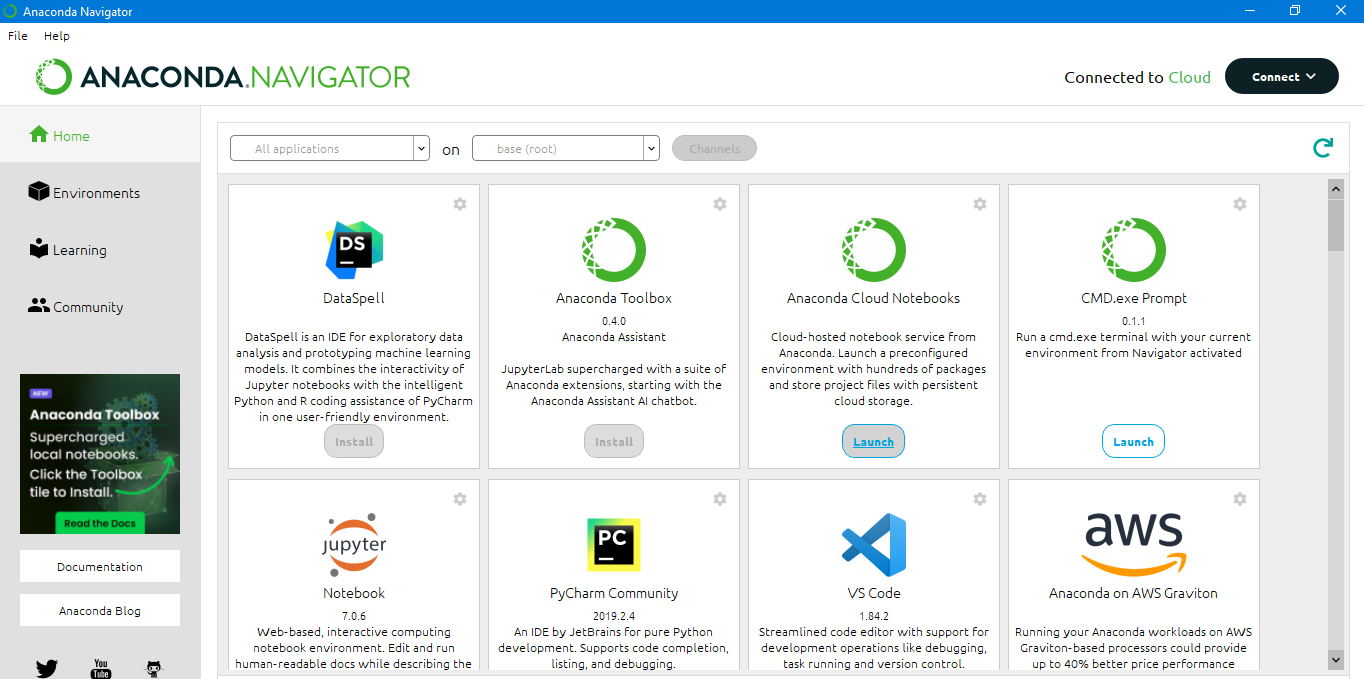


Figure 7: Anaconda Navigator

The Navigator includes all the code interpreter for python along with various other software or GUI to use in python programming. The “base(root)” in the navigation bar on the top of this GUI is the default environment for anaconda which in some cases should not be disturbed and for projects other environment should be created. Select the “Environments” option from the left navigation bar below the “file” option the following image would show.

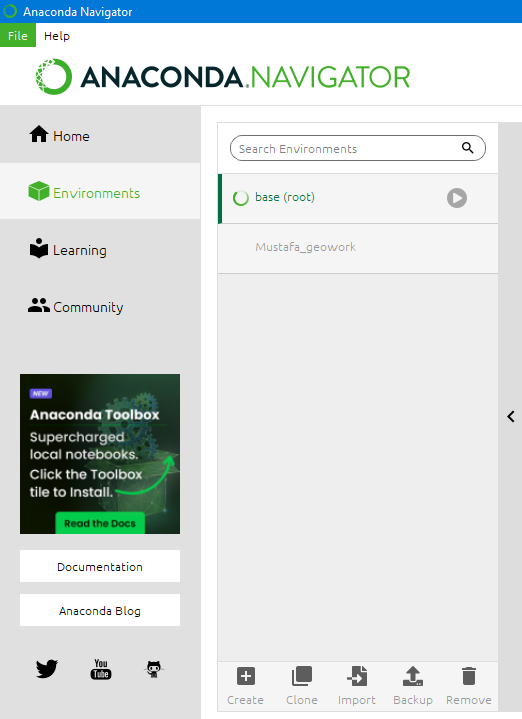


Figure 8: Creating Environments 1

Choose the create option on the bottom. The following pop up would show up

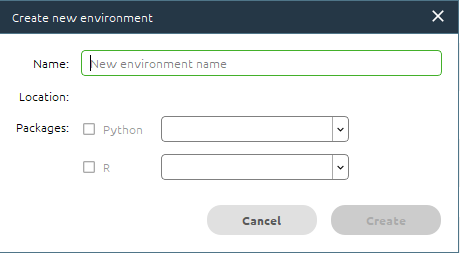


Figure 9: Creating Environment 2

Choose the name of the newly created environment and then choose the packages this environment would be composed of, after clicking on “create” the environment would be created. After creating the environment the next step would be to download the packages. Now the packages can be downloaded from the anaconda navigator or the conda prompt in the next section installation of packages from both conda prompt and anaconda navigator would be explained.

### **3.1.1 Installing Packages through Anaconda Navigator.**

The libraries can be installed through anaconda navigator by going to the environments and selecting your environment and then name the library to install on the right “search bar” and then click the check box next to the library and then click apply on the bottom as shown in the figure bellow:

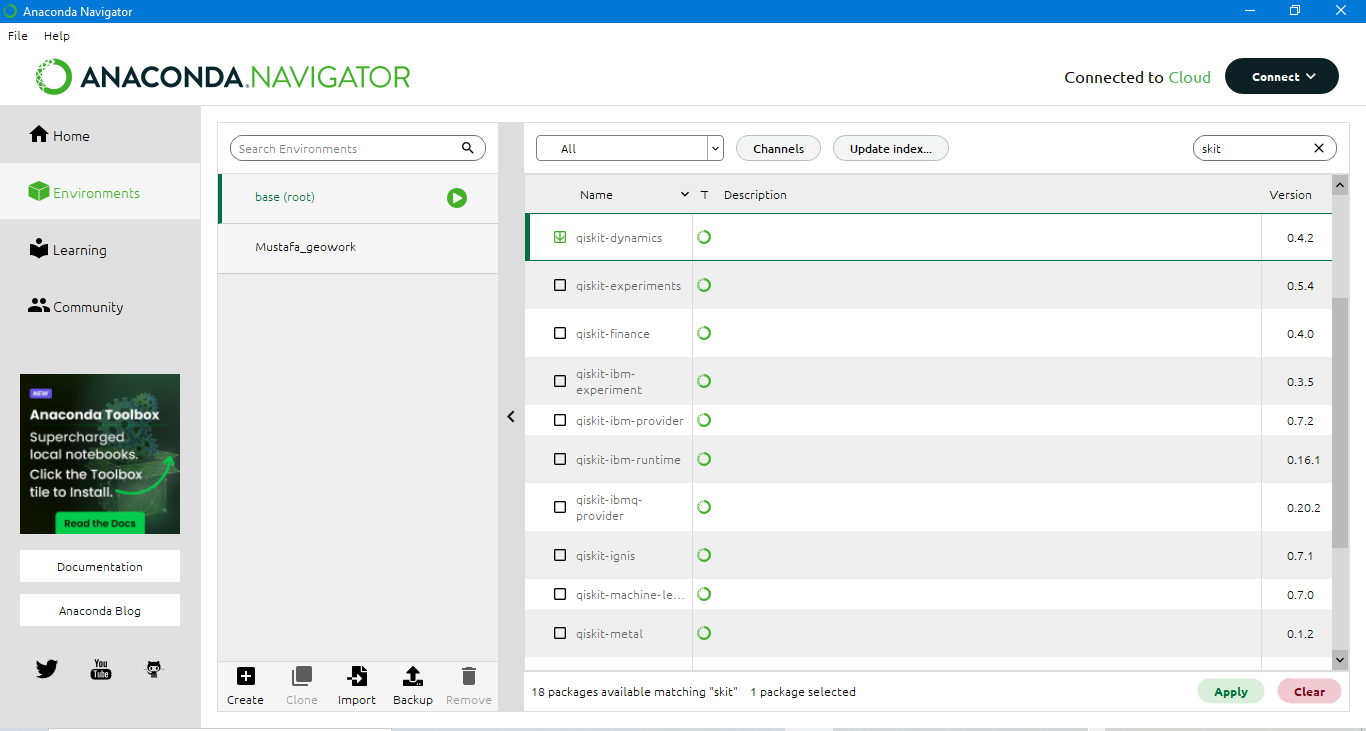


Figure 10: Installing Packages through anaconda navigator

After clicking on apply the package/ library would start installing on the environment the user chooses. This process would take time based on the internet connection.

### **3.1.2 Installing Packages through Anaconda Prompt.**

The anaconda prompt is somewhat similar to the cmd in windows but specifically for the anaconda environment. The conda prompt can be launched from the anaconda navigator as follows:

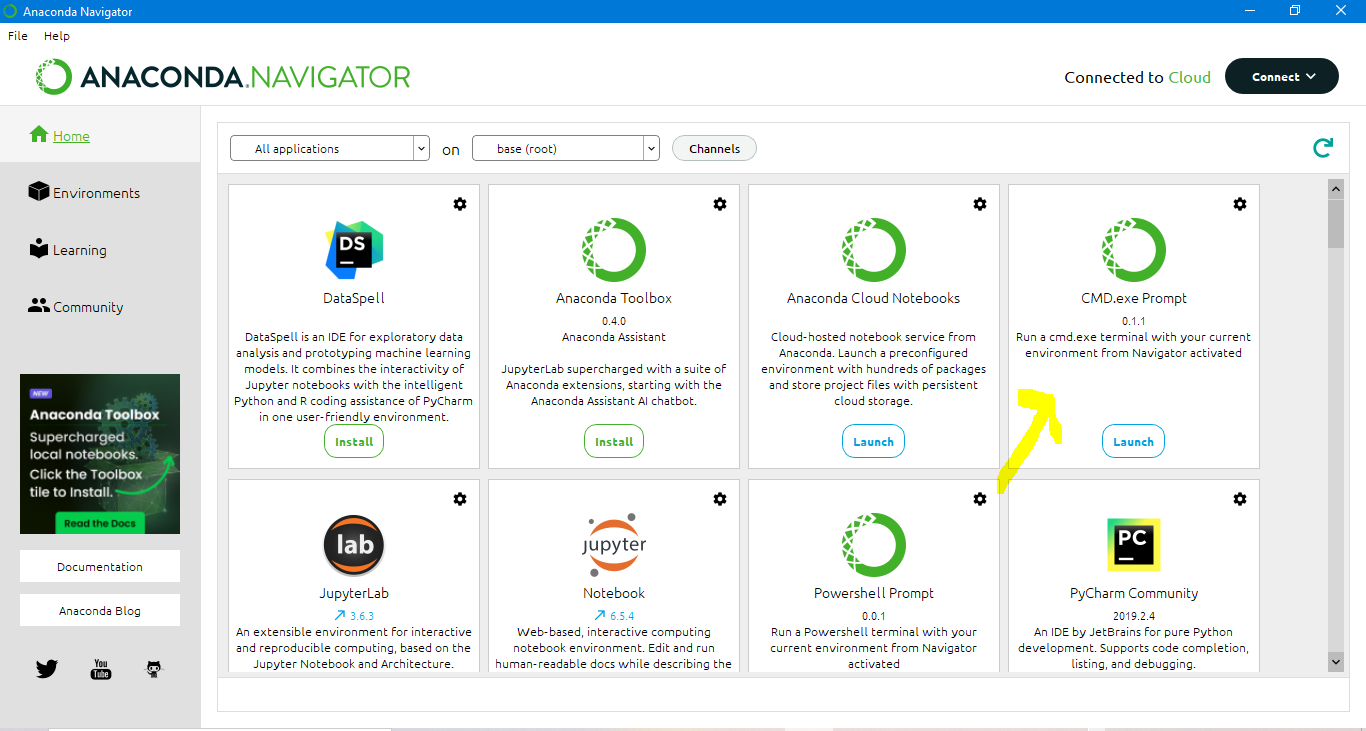


Figure 11: Starting Anaconda Prompt

After launching the prompt the following window would open up:

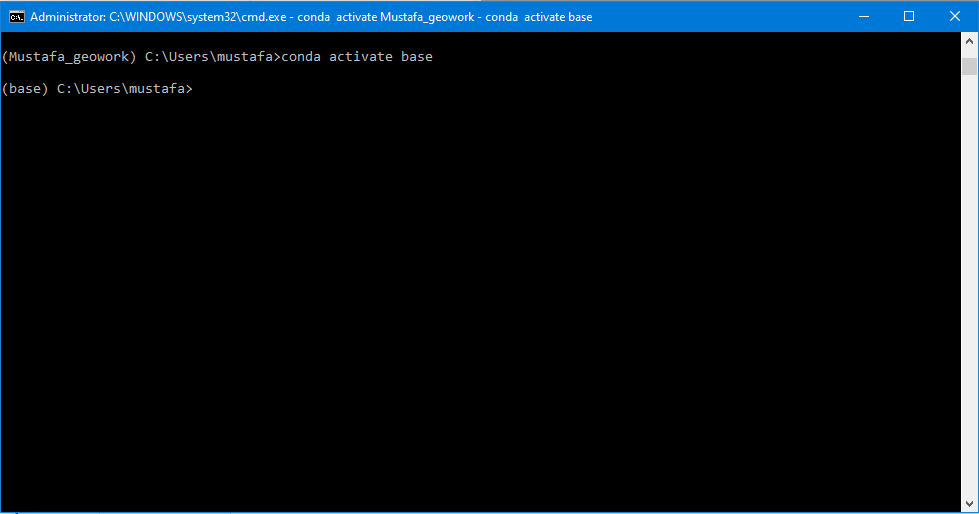


Figure 12: Anaconda Prompt

Now there are two ways to install the packages in anaconda prompt these two ways are as follows:

* Package installation through Conda install
* Package installation through Pip install

Both of these techniques have their own merits and demerits such as the pip install takes less time to install a package as compared to conda install but it might cause environmental issues, while on the other hand conda install would take more time to install a package but it would not lead to environmental issues. Let’s suppose we need to install satpy on anaconda the codes for installing satpy through conda install and pip install are as follows:

* **Pip install**
  + pip install satpy
* **Conda install**
  + Conda install –c conda-forge satpy (where “–c” refers to channel and “conda-forge” is a public channel) a channel is a repository where the packages would be fetched from or installed on.

## **3.2 Getting started with Satpy**

The basic functionality of satpy has already been explained in the **table 2 of section 3.** This package in simpler terms is maintained by the open source software group referred to as pytroll. Detailed overview of the usage of satpy is given below:

* This package handles multi-spectral satellite data from various sensors of geostationary and polar orbiting satellites. Satpy supports a lot of sensors such as NOAA AVHRR, HIMAWARI-8 AHI and SEVIRI sensors.
* Satpy can read and ingest satellite data in different formats through allowing users access to various tools which are used to manipulate data from various sensors. File formats such as HDF Netcdf and HRIT are supported by satpy.
* It can simplify data processing tasks such as Calibration, Navigation and Geolocation. It can converts said satellite data into various formats such as indices (NDVI, NDWI , etc.) trough band mathematics as it fetches the channel/ band information.
* It can help user to generate remote sensing products from satellite data.

### **3.2.1 Creating NDVI from HRIT data Of MSG**

This section deals with the usage of satpy in order to generate NDVI from HRIT data downloaded from EUMETSAT website. The coding interpreter used is **Jupyter Notebook in anaconda**. The first thing to do is to download the data. The data downloaded if uncompressed would include all channel files as follows:

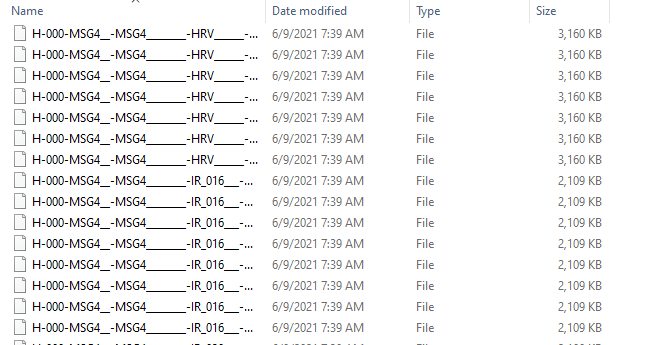


Figure 13: HRIT data files

**Step 1: Import the libraries**

The first step is to import the necessary packages for manipulating said data.

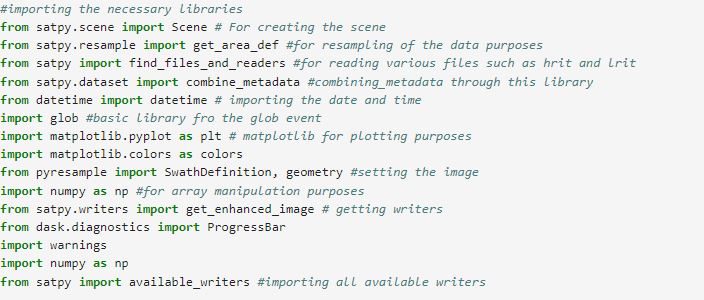
****

Figure 14: Importing the necessary libraries

**Step 2: Creating the scene (visualization scope)**

Since the data is uncompressed hence the glob module would be used to fetch said files as it creates a glob object (array) which includes all the HRIT files. The scene is creating by using the readers of satpy in this case the SEVIRI HRIT reader is used as shown below. “VIS006” and “VIS008” are channels which have been represented in the figure 1. For this project the scene was focused on South Africa. As shown below.

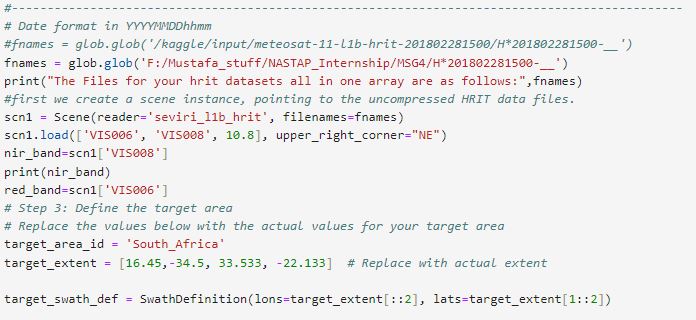
****

Figure 15: Creating the scene

**Step 3: Calculating NDVI**

**The NDVI which is an abbreviation for Normalized difference vegetation index is calculated by the following formula:**

NDVI= (NIR-RED)/(NIR+RED) where NIR is near infrared and RED is the visible band for the case of meteosat MSG HRIT data the channels were “VIS008” and “VIS006” their calculation is shown in the figure below . Matplotlib was the library used to plot the NDVI as shown below.

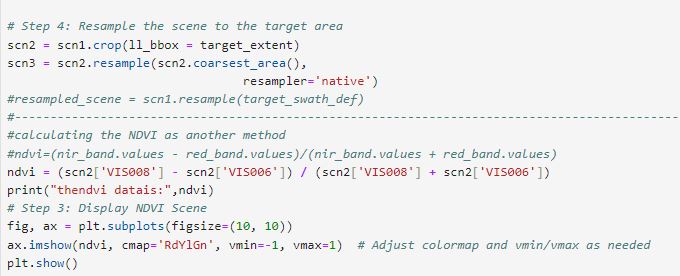
****

Figure 16: NDVI Calculation

**Step 4: The NDVI output shown through matplotlib**

The output shown by matplot lib is as follows:

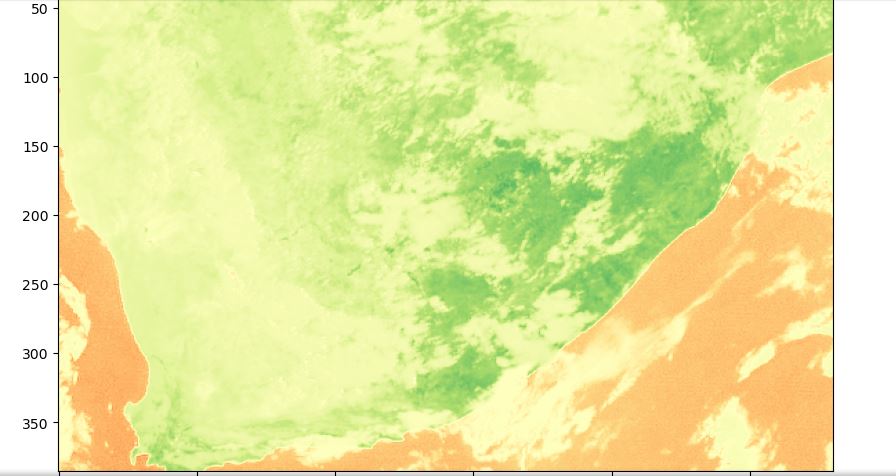


Figure 17: NDVI result through matplotlib

The green portion represents vegetation while the brown patches represents a lack of vegetation.

**Step 5: The NDVI output converted into Geo-tiff format**

For using the above output NDVI for further analysis it needs to be converted in a typical raster format referred to as Geotiff which can be done through satpy as it can also write data, satpy has inbuilt writers (the writer used for this case was the geotiff writer) as shown in the code below.

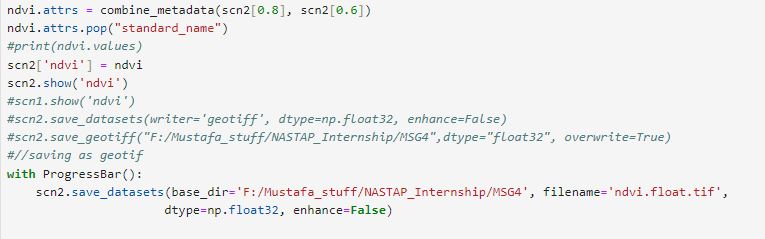


Figure 18:NDVI as geotiff

Then this NDVI geotiff can be used in software such as ArcMap and QGIS for this project ArcMap was used to visualize the NDVI geotiff as follows:

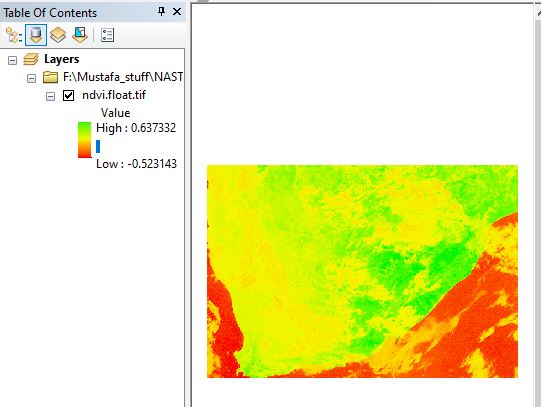


Figure 19: NDVI as geotiff in ArcMap

The values of NDVI are in the range of -1 till +1, values near to -1 represent a lack of vegetation while values closer to +1 represents vegetation.

### **3.2.2 Creating Fog Composite from HRIT data Of MSG**

In this case a fog composite was created from HRIT data (which was compressed). As already mentioned satpy can deal with compressed and uncompressed data. The working of this section is represented as follows:

**Step 1: Import the libraries**

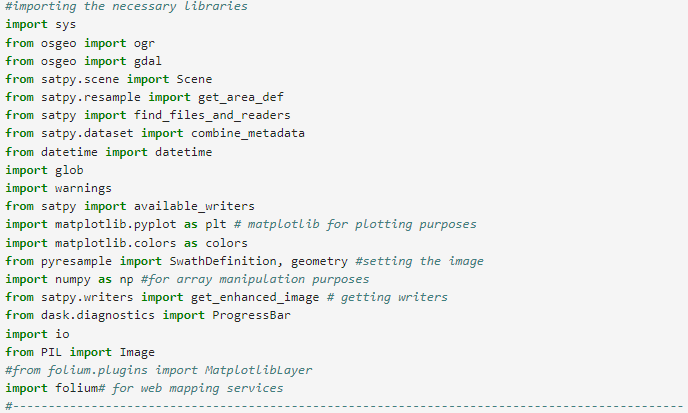


Figure 20: Importing libraries for fog

**Step 2: Creating the scene (visualization scope) for fog**

In order to create the scene for this case the data was read by the native SEVIRI reader of satpy as shown below:

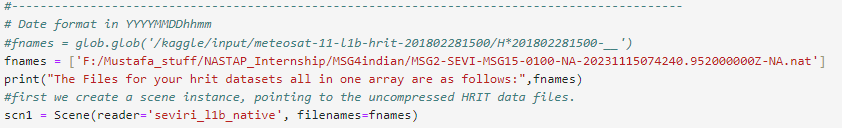


Figure 21: Creating the Scene for Fog

**Step 3: Cropping the scene with respect to the target area and generating the fog composite**

For this case the target area to be specified was the Indian Ocean specifically the area of Pakistan. Satpy has cropping capabilities with the scene.crop command. Satpy also has some inbuilt composites which are created through internal band/channel mathematics which are shown as below.

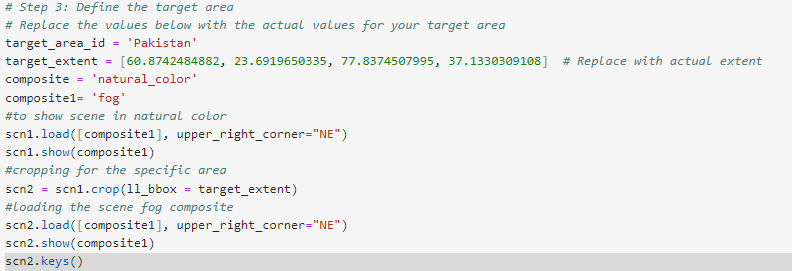


Figure 22: Creating the Fog composite

The output is as follows:

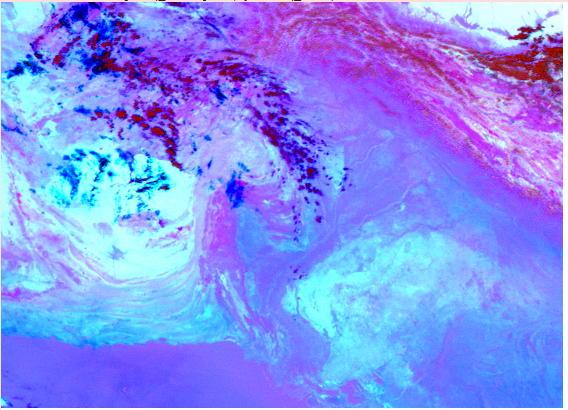


Figure 23: The Fog Composite

**Step 3: Exporting the Fog composite as a tiff image and displaying it on ArcMap**

The code used for exporting the scene result as a tiff image are as follows:

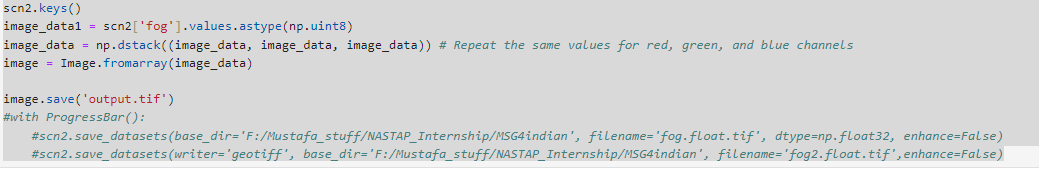


Figure 24: Exporting the scene as a tiff

The output displayed in ArcMap is as follows:

