Mustafa bokhari

Gis Intern

National Aerospace Science and Technology Park (NASTP)

Understanding the GRIB file format and Generating Product from said files

12/10/2023

Table of Contents

[**List Of Figures** 2](#_Toc153146891)

[**1.** **Introduction** 3](#_Toc153146892)

[**2. Downloading data** 4](#_Toc153146893)

[**3. Managing the python libraries for this project** 5](#_Toc153146894)

[**3.1 Getting started with Anaconda** 8](#_Toc153146895)

[**3.1.1 Installing Packages through Anaconda Navigator.** 10](#_Toc153146896)

[**3.1.2 Installing Packages through Anaconda Prompt.** 11](#_Toc153146897)

[**3.2 Getting started with GRIB data** 12](#_Toc153146898)

[**3.2.1 Reading GRIB files with Pygrib and generating Graphs through Matplotlib.** 12](#_Toc153146899)

[**3.2.2 Reading GRIB files through Pygrib and generating map plots with cartopy** 15](#_Toc153146900)

# **List Of Figures**

[Figure 1: Anaconda download 6](#_Toc153146910)

[Figure 2: Anaconda cloud 6](#_Toc153146911)

[Figure 3: Anaconda Installation Path 7](#_Toc153146912)

[Figure 4: Anaconda Installation Environment Path 7](#_Toc153146913)

[Figure 5: Anaconda Installation Complete 8](#_Toc153146914)

[Figure 6: Anaconda Navigator 9](#_Toc153146915)

[Figure 7: Creating Environments 1 9](#_Toc153146916)

[Figure 8: Creating Environment 2 10](#_Toc153146917)

[Figure 9: Installing Packages through anaconda navigator 10](#_Toc153146918)

[Figure 10: Starting Anaconda Prompt 11](#_Toc153146919)

[Figure 11: Anaconda Prompt 11](#_Toc153146920)

[Figure 12: GRIB data file 12](#_Toc153146921)

[Figure 13: Importing the necessary libraries 13](#_Toc153146922)

[Figure 14: Reading the GRIB file through Pygrib 13](#_Toc153146923)

[Figure 15: Accessing the GRIB data 14](#_Toc153146924)

[Figure 16: GRIB values representation through matplotlib. 14](#_Toc153146925)

[Figure 17: The Graph of U component of GRIB file through Matplotlib 15](#_Toc153146926)

[Figure 18: Importing libraries for GRIB file crtopy 16](#_Toc153146927)

[Figure 19: Creating a loop to read all the parameters within the GRIB files 17](#_Toc153146928)

[Figure 20: Creating map visualization of the GRIB data 17](#_Toc153146929)

[Figure 21: cartopy map visualization 18](#_Toc153146930)

[Figure 22: Exporting the cartopy plot as a tiff 18](#_Toc153146931)

[Figure 23: The cartopy plot as a raster 19](#_Toc153146932)

# **Introduction**

The Terminology GRIB is an abbreviation for “Gridded Binary” files which in simpler terminology are a type of data encoding in binary format , in general the data encoded within the GRIB files is meteorological, oceanographic and climate data. This kind of file format is used within environmental and atmospheric sciences for obtaining information regarding various atmospherically and climate parameters such as weather forecast models and climate models. These files are maintained by the World Meteorological Organization (WMO). There are various benefits of such file formats some of which are discussed below:

**Binary Format:** As already mentioned above the data stored in these GRIB files are in binary format which means that the data is machine-readable hence it is efficient in storing and transmitting information.

**Self-contained Messages:** The GRIB files have individual messages in human understandable terminologies which contain information about a specific parameter at a particular point. Basically these messages include information about parameters such as the name, unit, value and etc.

**Representation of Grids:** The general GRIB file supports various kinds of grid representations such as longitude-latitude grids and etc.

**Compression:** The GRIB files use various compression techniques to compress the data within them and hence making data transmission quite easy.

**Global Standard:** The GRIB files are data files with a global standard maintained by WMO

In general there are three versions of GRIB. The first version revered to as version 0 was used but to a limited extent and is no longer operational. Version 1 or also referred to as first edition is operationally used worldwide by most meteorological centers. A newer generation has been introduced known as GRIB second edition. This second edition is used for derived products distributed in the EUMETCAST of MSG.In GRIB2 (second addition) longitude values lie between 0 to 360 degrees, the data in GRIB2 is described in template/ table form.

# **2. Downloading data**

GRIB files in general can be downloaded from various websites but it can also be created. The following websites provide data in GRIB format and also various softwares with GUI’s to read and display GRIB data.

Table 1: GRIB data download

|  |  |  |
| --- | --- | --- |
| **Website name** | **Link to websites** | **Information** |
| Pivotel | https://www.pivotel.com/pivotel-free-weather-grib-files | This website provides free access to various GRIB files but the data in these GRIB files is restricted to Wind and Wave information of various regions. It does not provide any software or tool to manipulate said data. |
| OpenGribs | https://opengribs.org/en/downloads | This website does not provide any GRIB files but it does provide a software with tools to access and manipulate GRIB files |
| METEO-Consult | https://www.eumetsat.int/ordering-client | This website does provide access to GRIB files for Europe and Atlantic regions but a user has to register to said website in order to access its services. |

There are other methods of obtaining GRIB files the afro mentioned websites were just the tip of the iceberg.

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

The libraries necessary for the handling of GRIB data are as follows:

* Pygrib
* Gdal
* Matplotlib
* Rasterio
* Cartopy
* Geopandas

Table 2: Libraries

|  |  |
| --- | --- |
| **Library Name** | **Usage** |
| Pygrib | Pygrib in general as the name implies is a python library specifically designed for handling and working with GRIB files and data. This library makes handling meteorological and environmental data easy. It can allow users to access (read) GRIB files, accessing the first message etc. It allows users to extract various parameters from GRIB files, allows access to location (longitude and latitude) information within GRIB files. |
| 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 |
| Matplotlib | This library as the name implies is used to generate graphical plots of various data. |
| Cartopy | Cartopy is a python library designed for cartographic projections and geospatial data visualization and it provides tools for creating maps and etc. |
| Rasterio | This Library as the name implies is used to handle raster data formats, it also has the capabilities to convert various data formats to raster data types such as Geotiff |
| 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. |

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:

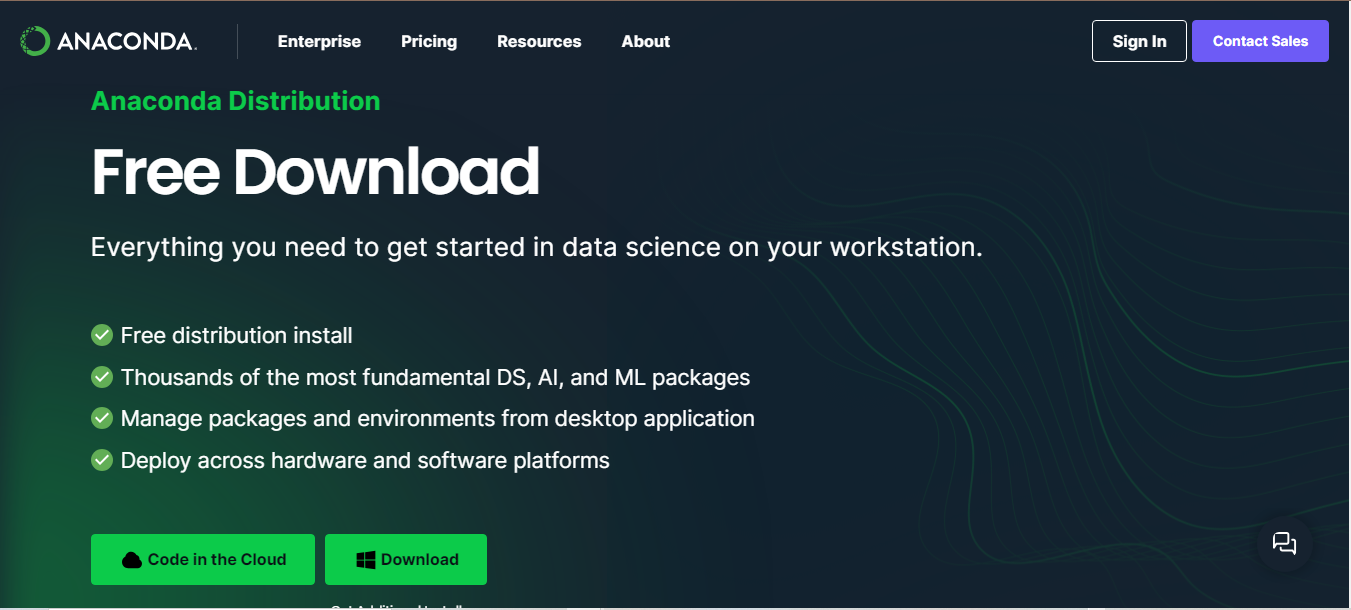


Figure 1: Anaconda download

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

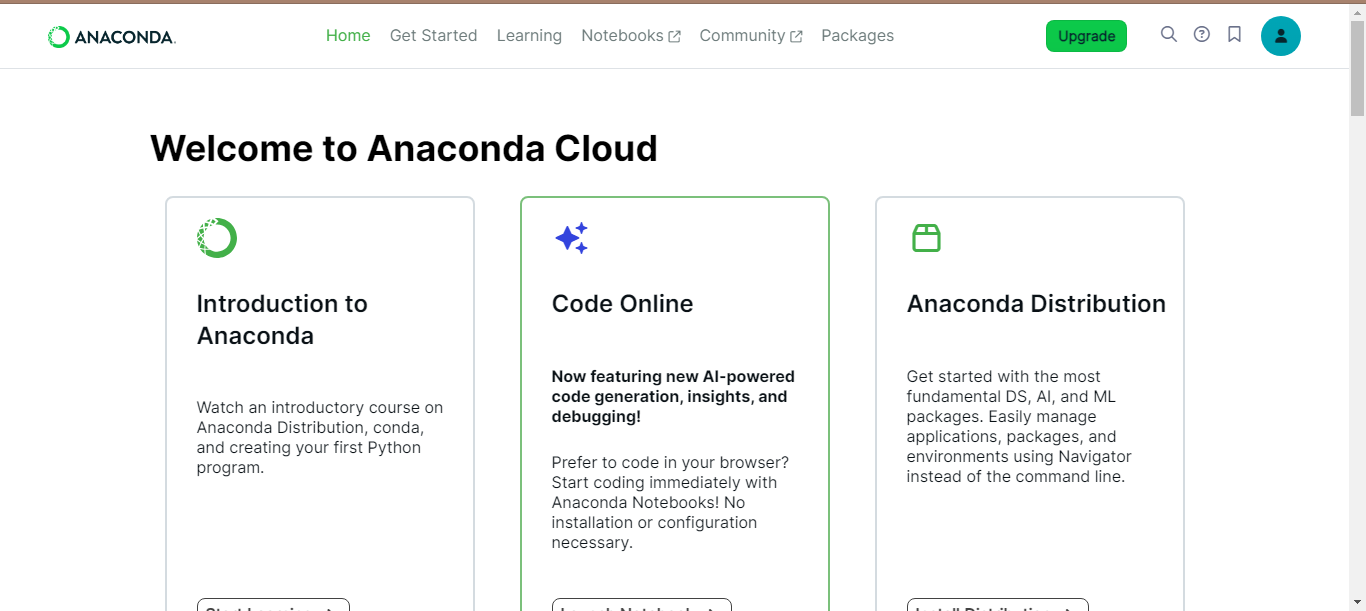


Figure 2: Anaconda cloud

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:

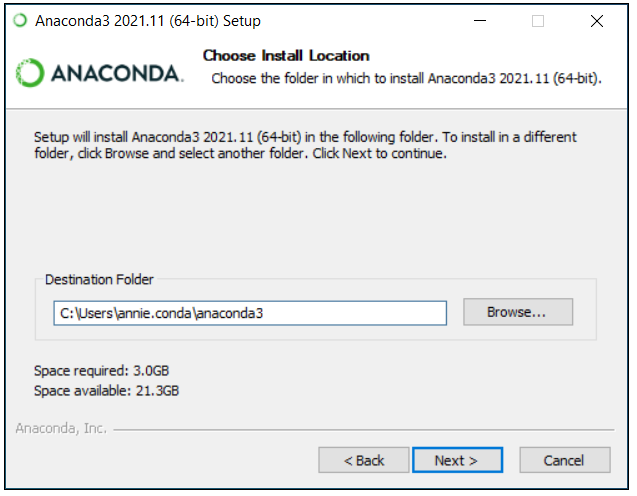


Figure 3: Anaconda Installation Path

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

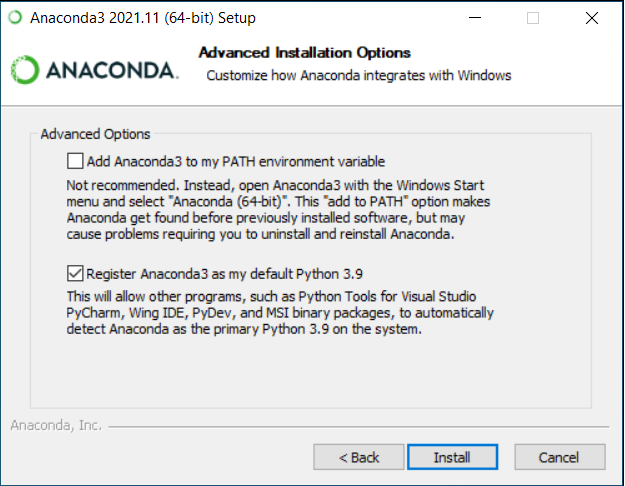


Figure 4: Anaconda Installation Environment Path

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:

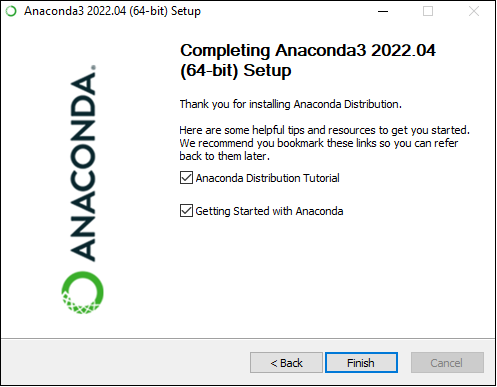


Figure 5: Anaconda Installation Complete

## **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

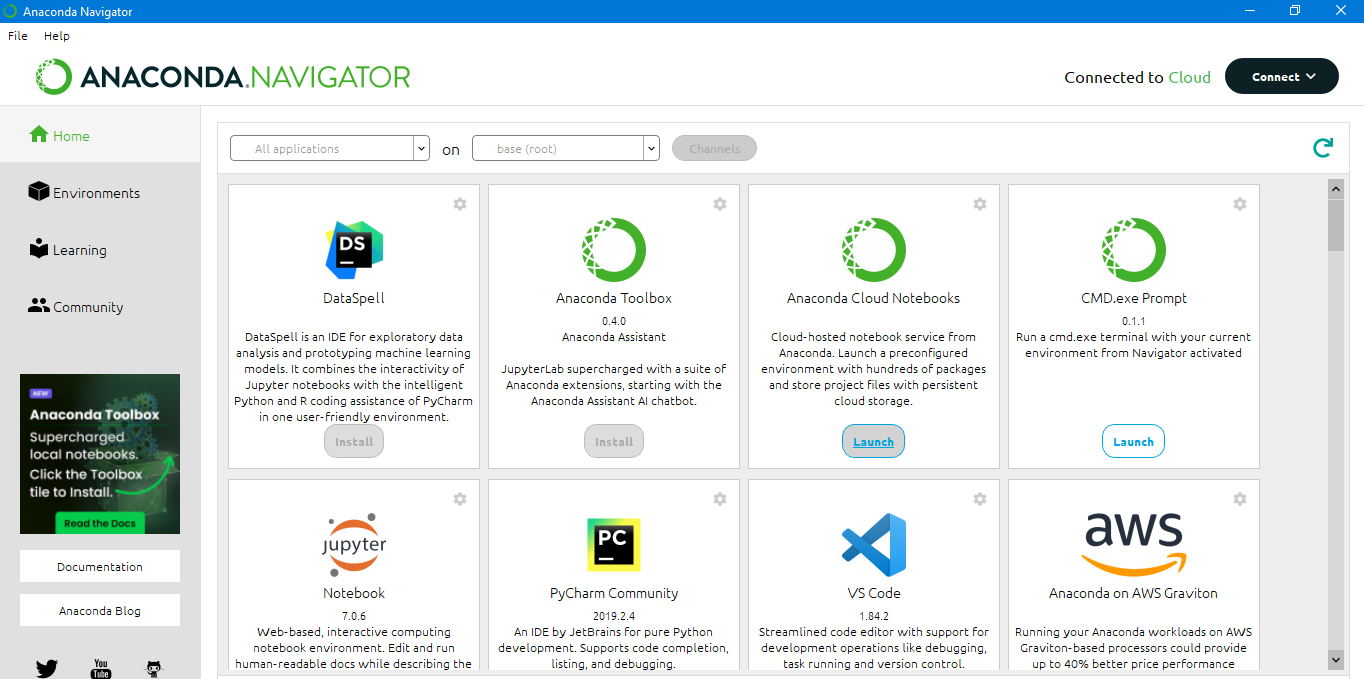


Figure 6: Anaconda Navigator

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.

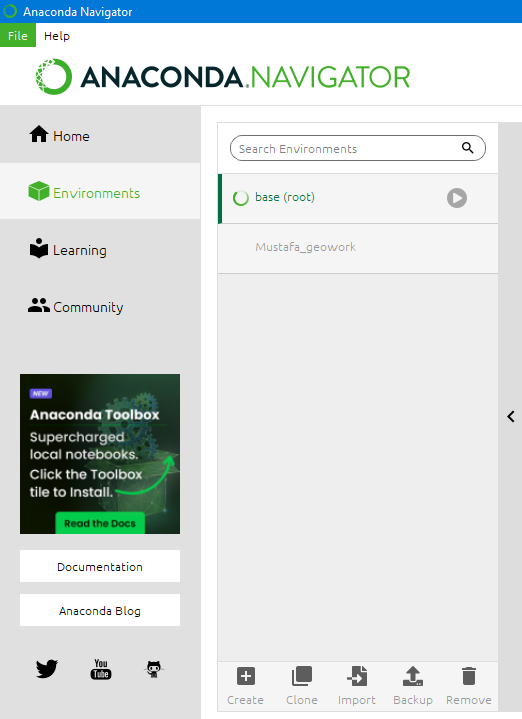


Figure 7: Creating Environments 1

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

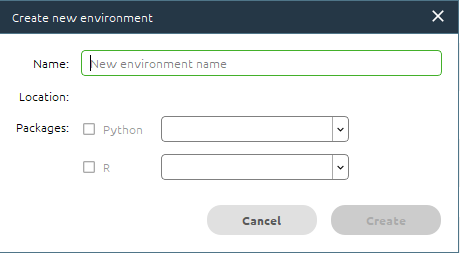


Figure 8: Creating Environment 2

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:

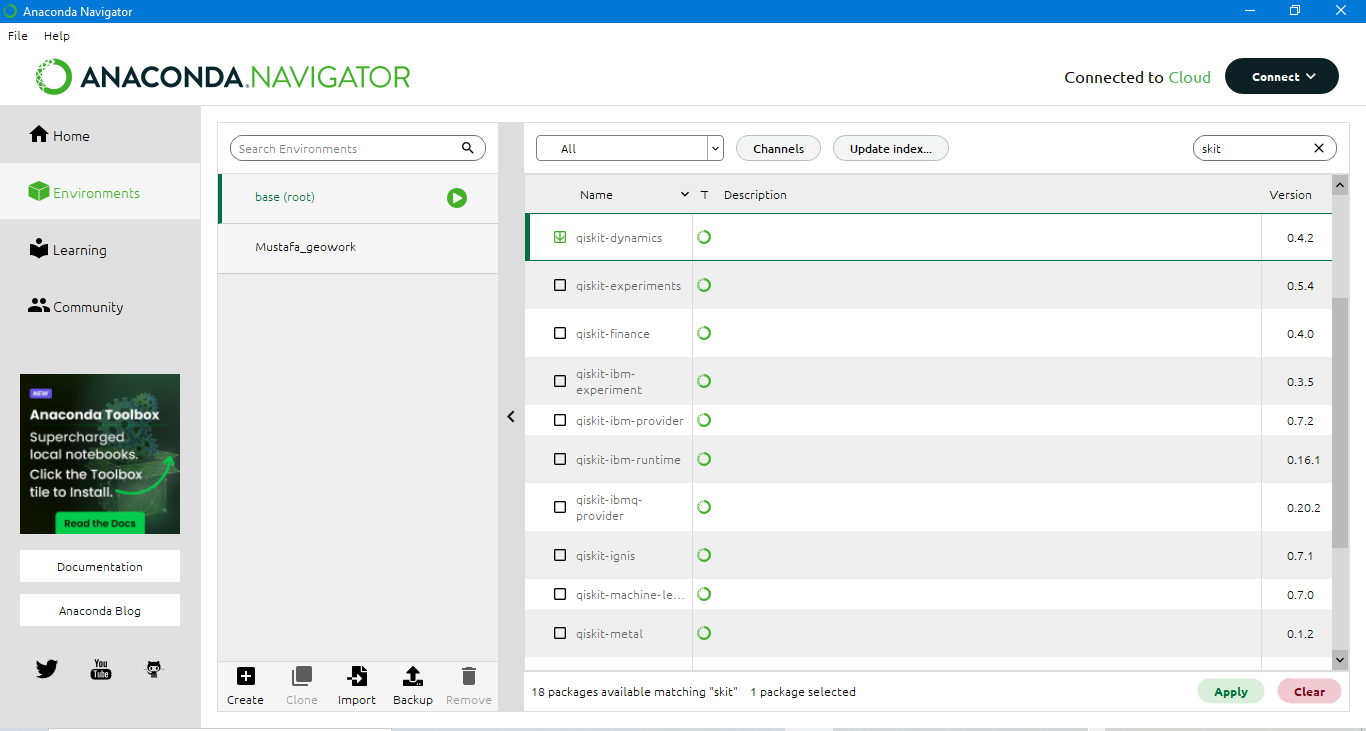


Figure 9: Installing Packages through anaconda navigator

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:

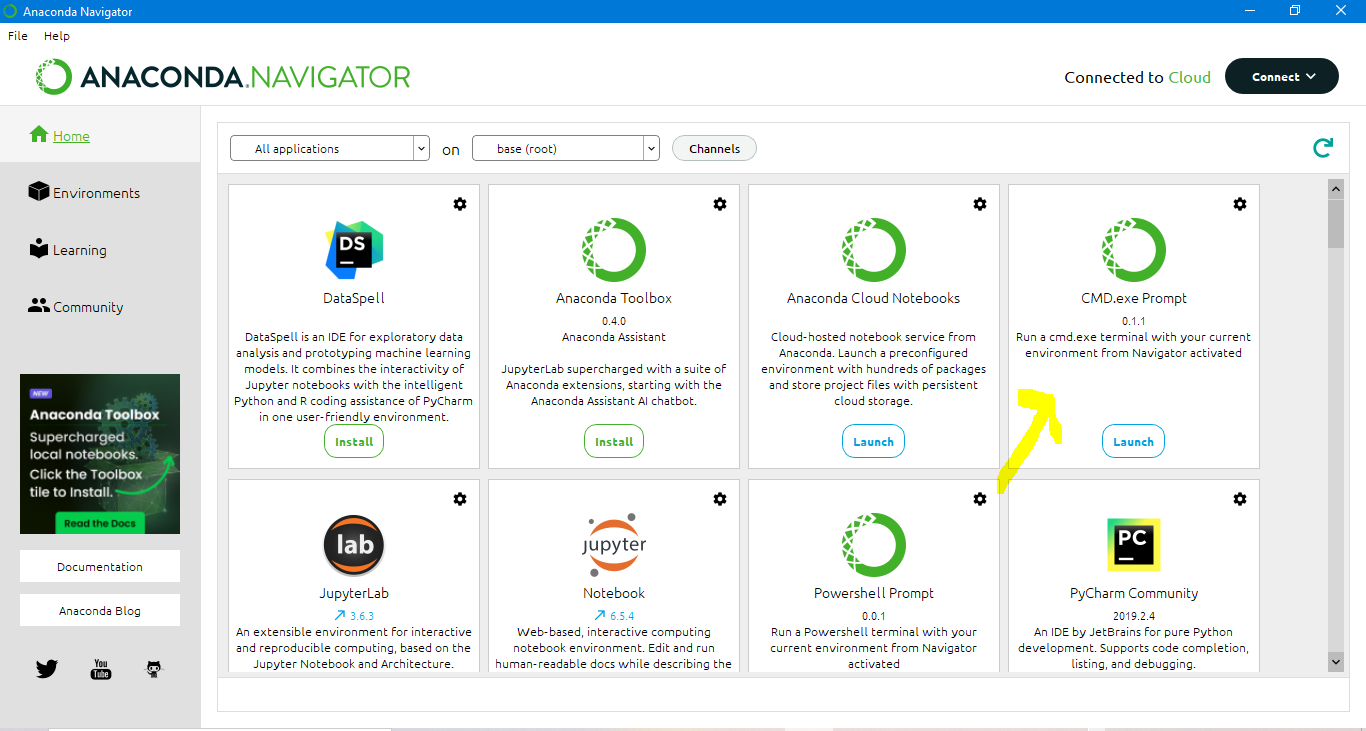


Figure 10: Starting Anaconda Prompt

After launching the prompt the following window would open up:

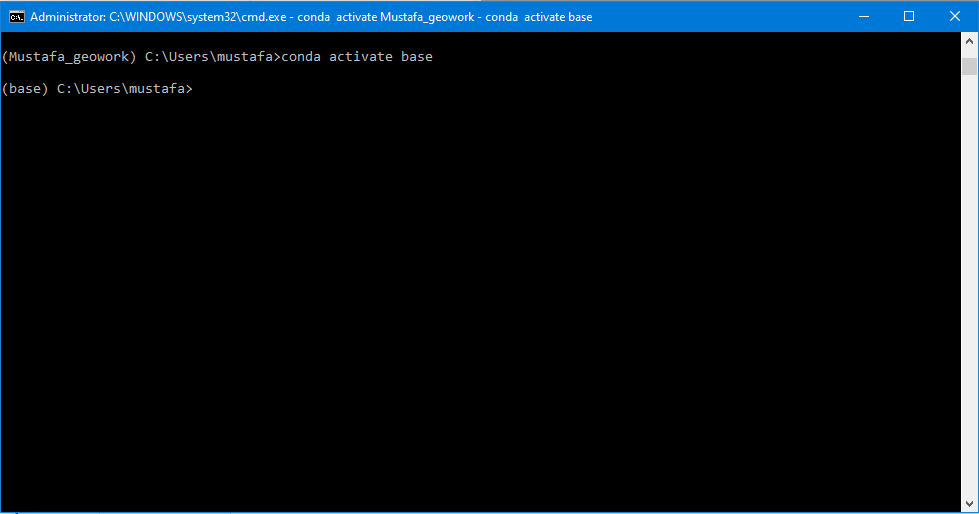


Figure 11: Anaconda Prompt

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 pygrib on anaconda the codes for installing pygrib through conda install and pip install are as follows:

* **Pip install**
  + pip install pygrib
* **Conda install**
  + Conda install –c conda-forge pygrib (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 GRIB data**

This section would deal with the handling of GRIB data. This section would be divided into 2 sections which are as follows:

* Reading GRIB files with Pygrib and generating Graphs through Matplotlib
* Reading GRIB files with Pygrib and generating Map plot through cartopy.

### **3.2.1 Reading GRIB files with Pygrib and generating Graphs through Matplotlib.**

This section deals with the use of Pygrib to read the GRIB files. The GRIB file used for this study is as follows:

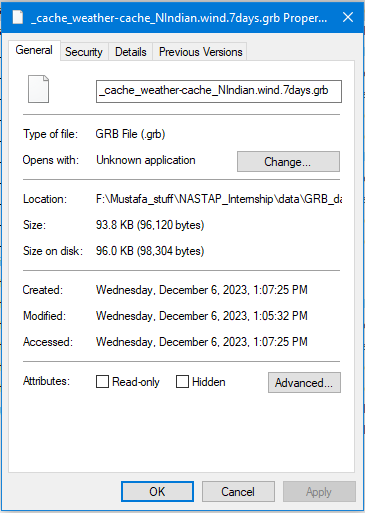


Figure 12: GRIB data file

**Step 1: Import the libraries**

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



Figure 13: Importing the necessary libraries

**Step 2: Reading the GRIB file**

After importing the necessary libraries the first thing to do is to reade the GRIB file through the use of Pygrib. If the file is correct then the script would run without any error as follows:

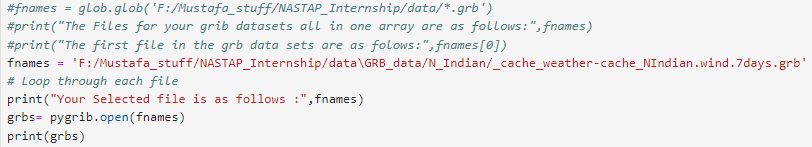


Figure 14: Reading the GRIB file through Pygrib

**Step 3: Fetching the Data within the GRIB file.**

As the GRIB file contains information about various atmospherically and environmental parameters, there are two ways to access the data within. Data regarding a single parameter can be accessed if a user knows about the information stored in the GRIB files, however if the user has no to little information regarding the data within the GRIB file then an automated loop can be created to read all the data within the GRIB files. For this study a loop was created to read all the data within the GRIB file as follows:

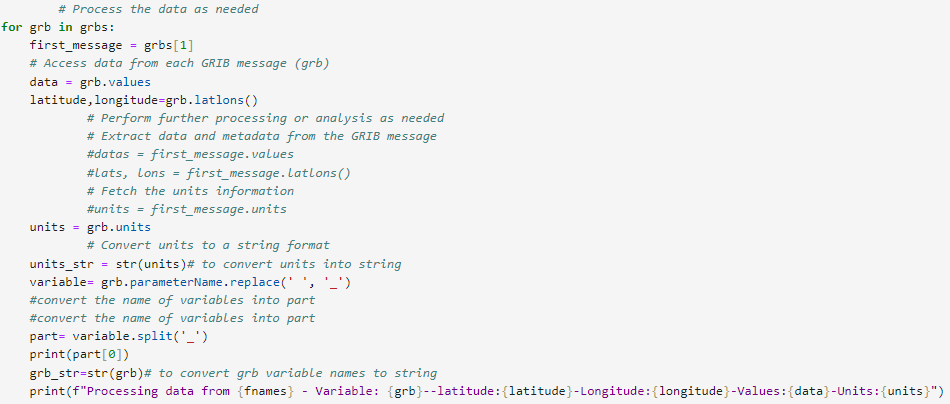


Figure 15: Accessing the GRIB data

As it can be observed from the figure above information such as longitude, latitude, units and value of the parameters stored in the GRIB file can be accessed and this information is printed in the output in the print command as shown in the figure above

**Step 4: Plotting the values of the GRIB data through matplotlib with units**

Matplotlib is the library used to provide the representation of the GRIB data in the form of a Graph in terms of the longitude and latitude values

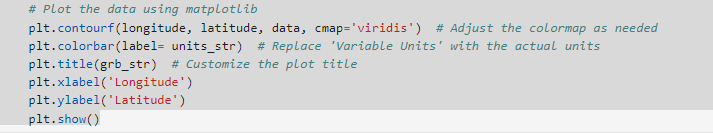


Figure 16: GRIB values representation through matplotlib.

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

**Step 5: The Graphical representation of the GRIB data**

In this study the GRIB file was obtained from pivotel website and the data contained in this GRIB file was the wind information of the **north-Indian ocean region,** the components of wind represented were the U and V component as follows:

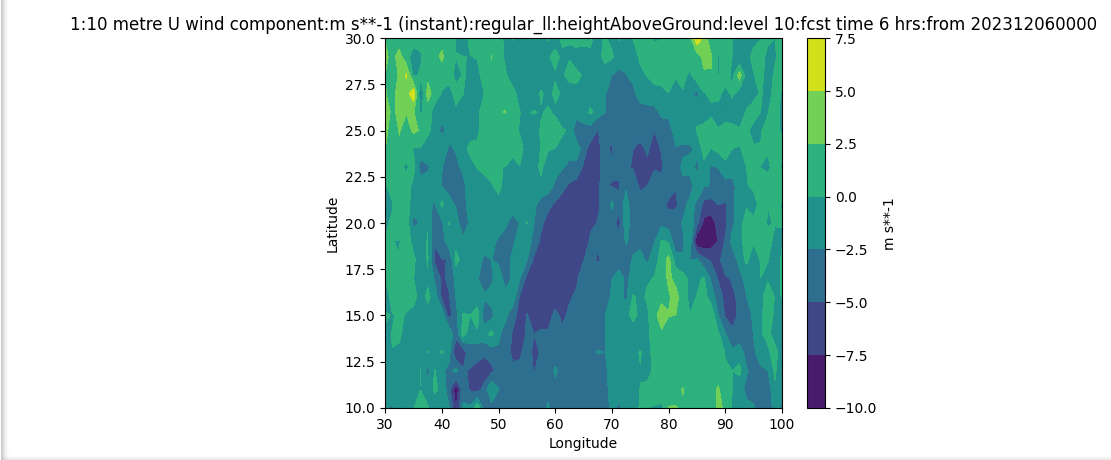


Figure 17: The Graph of U component of GRIB file through Matplotlib

### **3.2.2 Reading GRIB files through Pygrib and generating map plots with cartopy**

In this section the Same GRIB file is read through Pygrib but instead of graph a map representation of GRIB data is generated through cartopy and then the map is converted into a raster containing the information through rasterio.

**Step 1: Import the libraries**

The first thing to do is to import the necessary libraries , the most important libraries in this case are the pygrib, numpy, cartopy and rasterio libraries.



Figure 18: Importing libraries for GRIB file crtopy

**Step 2: Reading the GRIB File data and creating a loop.**

Reading the GRIB file data through pygrib and then generating a loop to access every parameter within the GRIB file as follows:

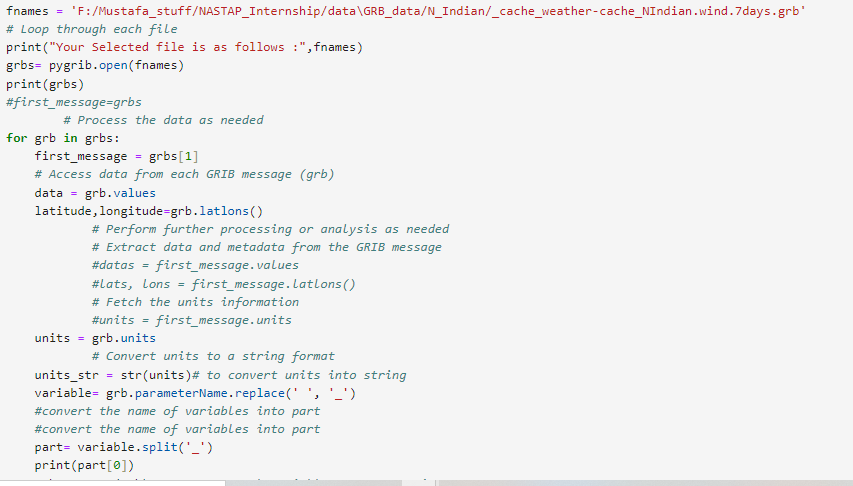


Figure 19: Creating a loop to read all the parameters within the GRIB files

**Step 3: Creating the map visualization of the GRIB data through Cartopy.**

In this case cartopy was used to generate the map type visualization of the GRIB data through using the longitude and latitude information provided in the GRIB file. The wind values were used to generate the thematic map and the colorbar legend.

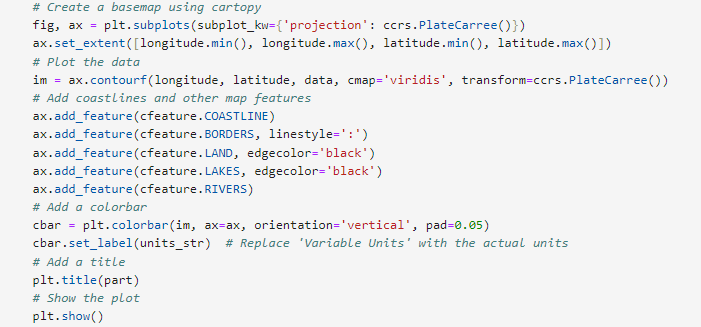


Figure 20: Creating map visualization of the GRIB data

The output visualization of a parameter component is as follows:

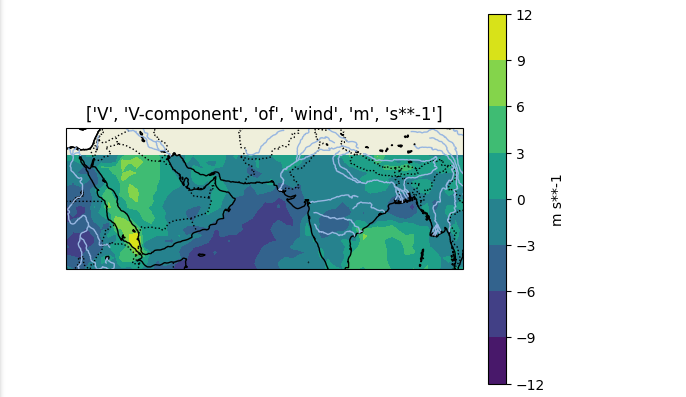


Figure 21: cartopy map visualization

**Step 3: Exporting the cartopy plot as a tiff image and displaying it on ArcMap**

The code used for exporting the cartopy plot as a tiff image are as follows:

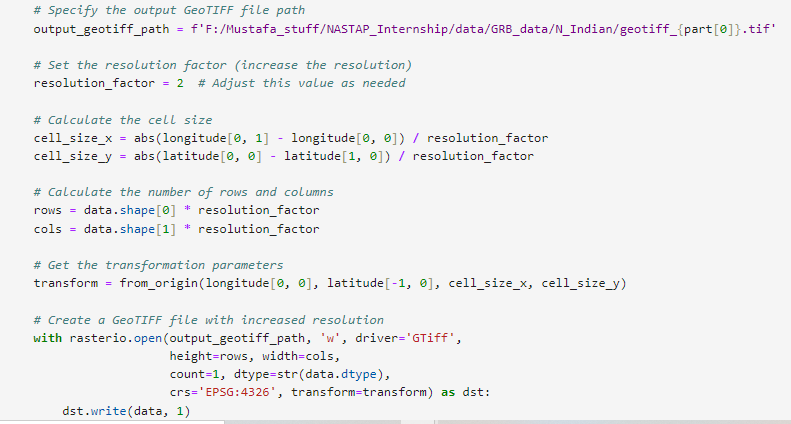


Figure 22: Exporting the cartopy plot as a tiff

The output displayed in ArcMap is as follows:

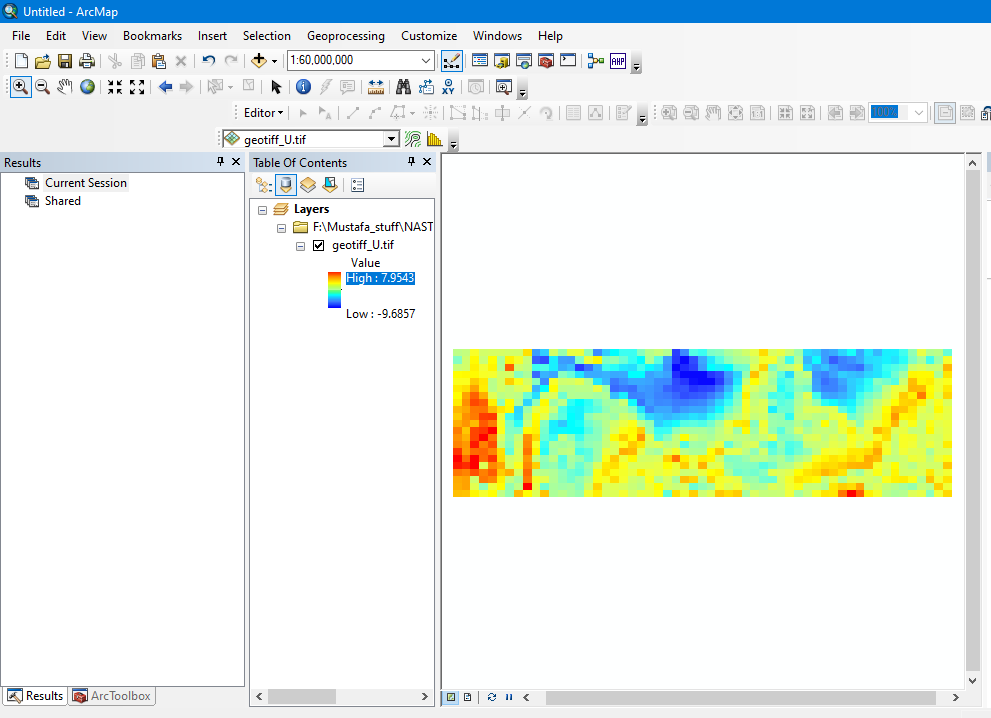


Figure 23: The cartopy plot as a raster