

# ITURPropagPy Quick Guide

## Contents

1. Python Installation .....	2
1.1 Python Packages Installation .....	2
1.2 ITURPropagPY Installation .....	4
1.2.1 Install from GitHub .....	4
1.2.2 Install from pip stream .....	6
2. Define Python Interpreter .....	6
3. Python Script .....	7
3.1 Atmospheric Attenuation .....	8
3.1.1 Single location vs single probability .....	8
3.1.2 Single location vs different probability .....	12
3.1.3 Single location vs different probability and plotting output .....	14
3.2 Export data .....	17
3.2.1 Export data to CSV file: .....	18
3.2.2 Export data to text file: .....	19
3.2.3 Export data to Excel file .....	20
3.2.4 Export data to h5 file: .....	21

# 1. Python Installation

There are two ways to download and install the source of Python software. (a) Download the main core of Python software in this link <https://www.python.org/downloads/> or (b) download the anaconda distribution from the link <https://www.anaconda.com/distribution/>. The anaconda distribution nearly contains all of the python packages that after installation there is no need to install different packages manually. Also, it is strongly suggested to install the open source Visual Studio Code software from the link ( <https://code.visualstudio.com/> ), which is the best for editing and running the code scripts in different programming languages. It should be noticed that the anaconda has it's own platform for editing codes called *spyder*. The picture of both VSCode and spyder can be seen in Figure 1-1 and Figure 1-2.

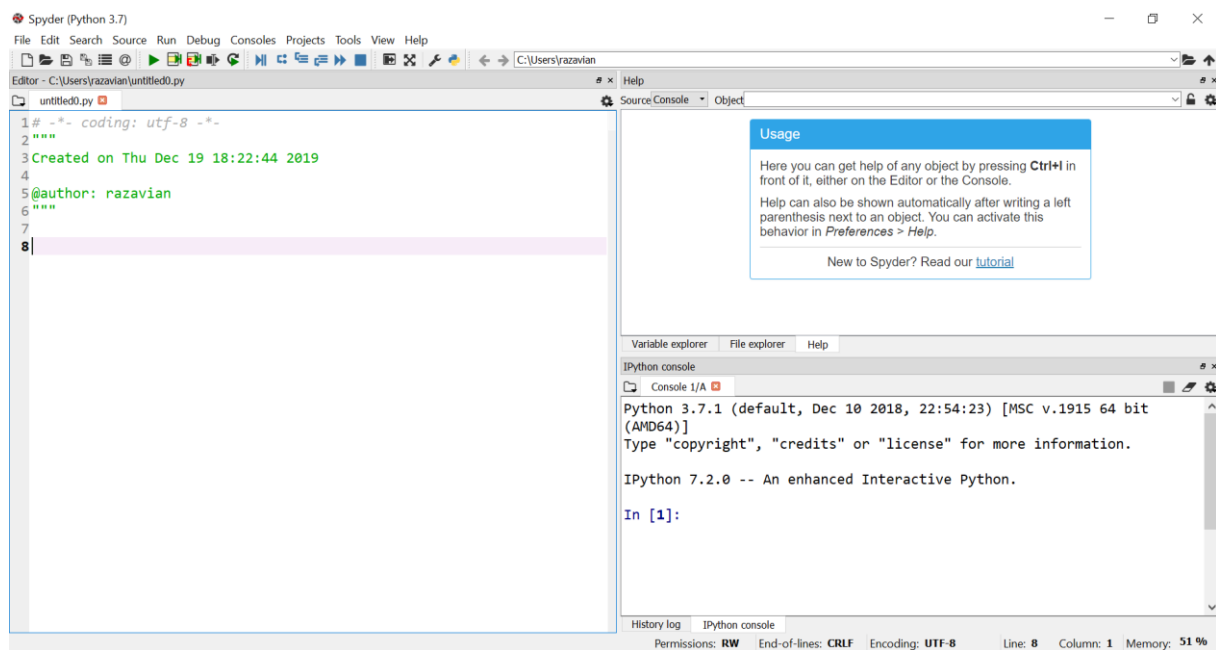


Figure 1-1: spyder software

## 1.1 Python Packages Installation

For most of application, using python installing different packages are necessary. Also, for the *ITURPropagPY* the packages “*scipy*”, “*numpy*”, “*matplotlib*”, “*joblib*”, “*pandas*”, “*h5py*”, “*astropy*”, “*pyproj*”, and “*basemap*” are necessary. If the *anaconda* distribution software is installed all of these packages (except “*basemap*”) are installed automatically and no need to install them again. However, here we will show how to install them one by one. Open the VSCode software as in Figure 1-2 and from “*File → Open Folder*” open a folder as you want. Then from the tab “*Terminal*” open a “*New Terminal*”.

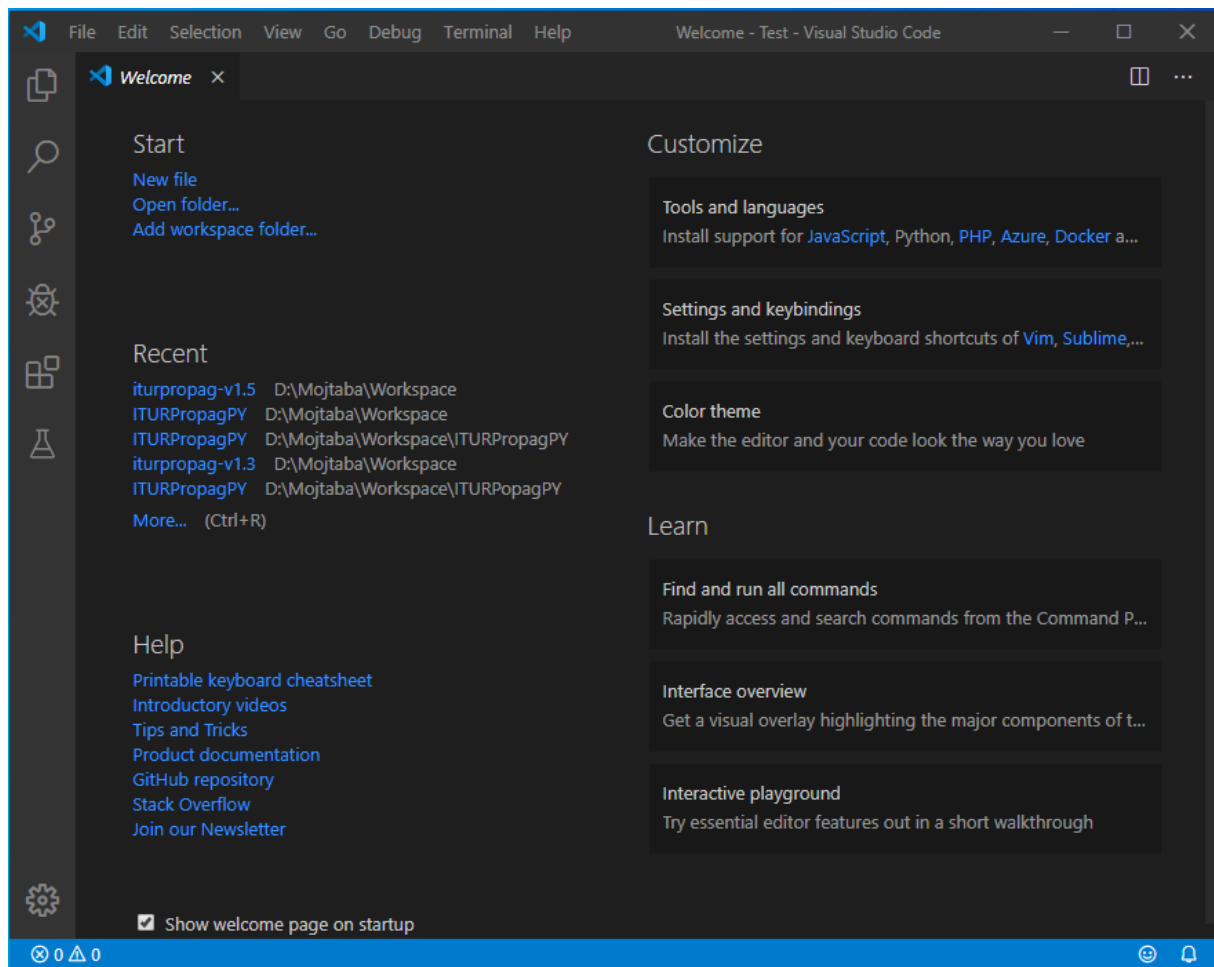


Figure 1-2: VSCode Software

In the opened terminal write the code command below and then enter.

```
$ pip install scipy
```

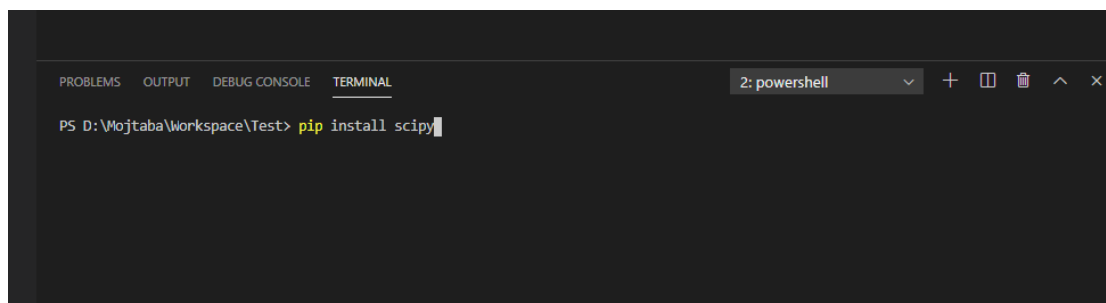


Figure 1-3: Install the packages for python

For all of the packages follow exactly the same procedure one by one or it can be done in one single command line as follow :

```
$ pip install scipy numpy matplotlib joblib h5py pyproj astropy pandas
```

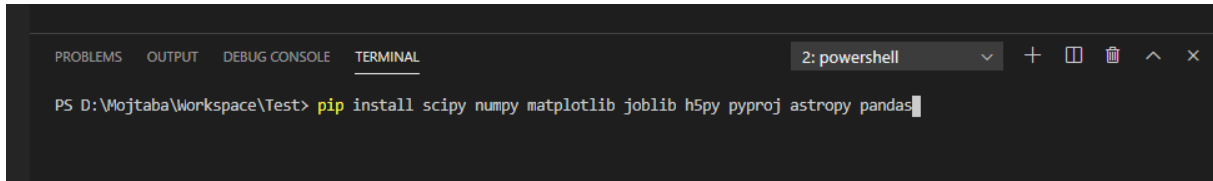


Figure 1-4: Install multiple packages for python in one single command line

Installing the “*basemap*” package is more complex than installing the other packages. If the anaconda distribution software is installed, the “*basemap*” can be installed more easily by following the command line below (<https://anaconda.org/anaconda/basemap>):

```
$ conda install -c anaconda basemap
```

If “*anaconda*” is not installed, for installing “*basemap*” the compilers for C/C++ will be needed. The full procedure is explained in the source page of “*basemap*”:

<https://matplotlib.org/basemap/users/installing.html>

It should be noted that installing the “*basemap*” is not crucial for using the *ITURPropagPY* package. Only if you want to show the attenuation on the map it will be needy.

## 1.2 ITURPropagPY Installation

To install ITURPropagPY you can do it in two ways: (a) the main source files from GitHub, (b) Use pip stream.

### 1.2.1 Install from GitHub

Source files of *ITURPropagPY* files can be download from the webpage on GitHub (Figure 1-5):

<https://github.com/mrazavian/ITURPropagPY>

After downloading, extract the zip file and from the *VSCode* software open the entire folder. Then from the tab “*Terminal*” open a “*New Terminal*” and run below command lines respectively:

a) First, to be sure the latest versions of *setuptools* and *wheel* are installed run (Figure 1-6):

```
$ python -m pip install --user --upgrade setuptools wheel
```

b) The next step is to generate distribution packages for the package (\*.whl and \*.tar.gz files). (Figure 1-6)

```
$ python setup.py sdist bdist_wheel
```

This command will create two files “*iturpropag-1.5-py2.py3-none-any.whl*” and “*iturpropag-1.5.tar.gz*” in the “*dist*” folder

c) Now it’s time to install the “*iturpropag-1.5-py2.py3-none-any.whl*” file using pip command (Figure 1-7)

```
$ pip install dist/iturpropag-1.5-py2.py3-none-any.whl
```

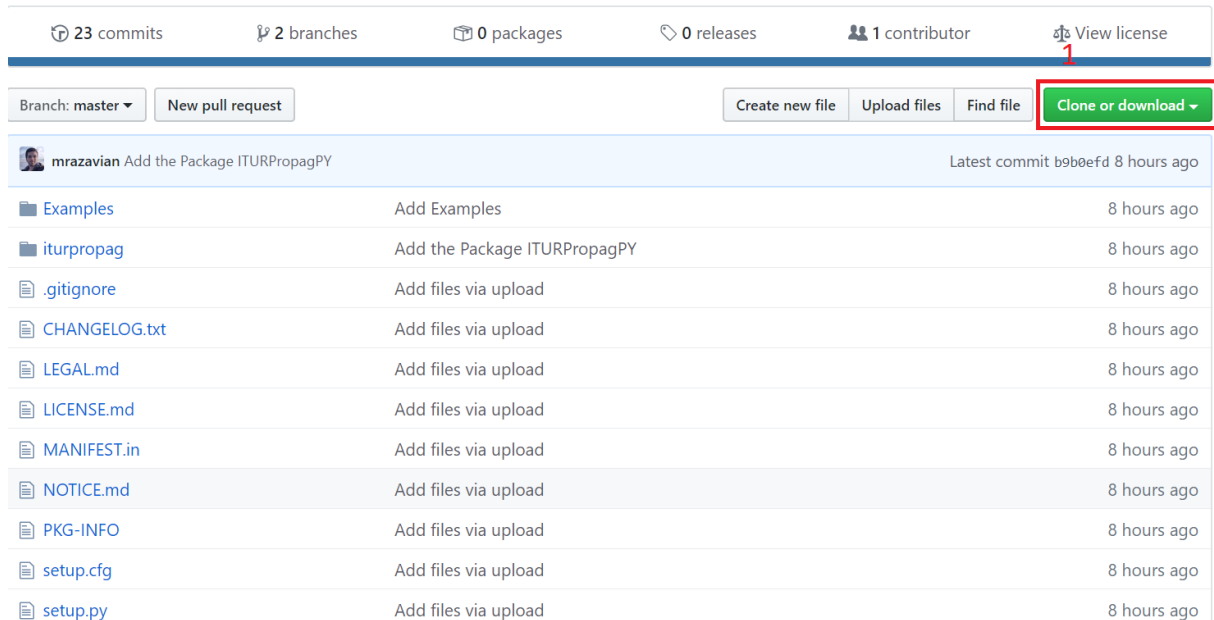


Figure 1-5: Source files of ITURPropagPY package on GitHub

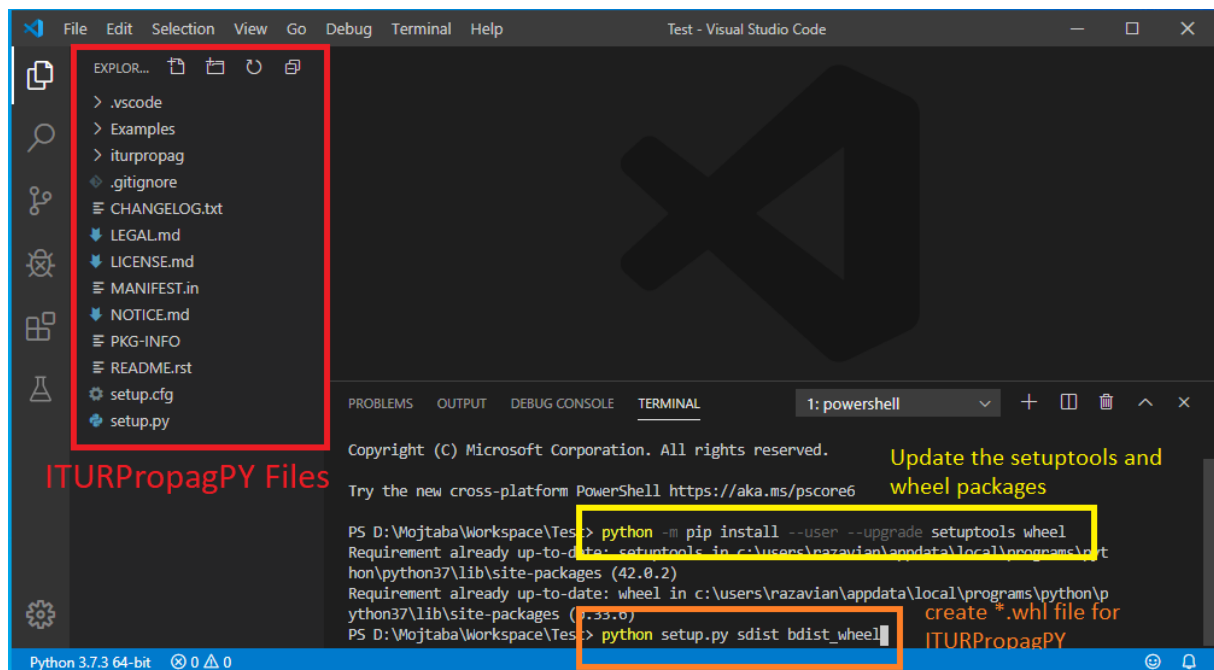


Figure 1-6: Create ITURPropagPY wheel file

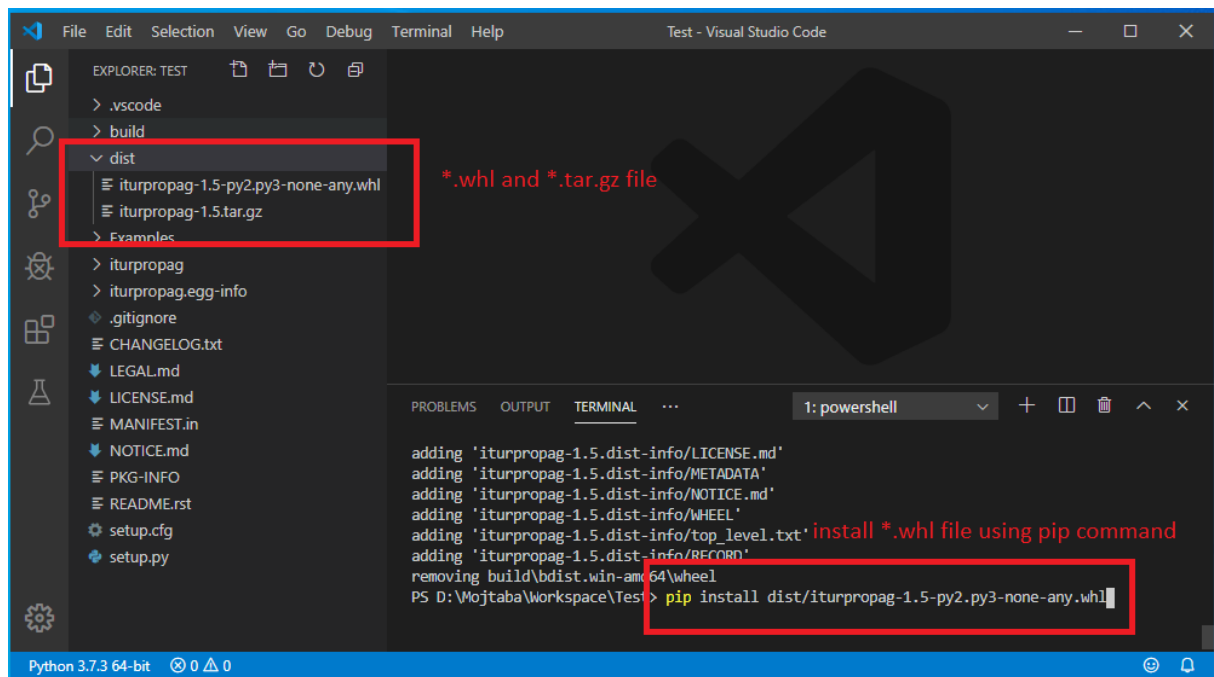


Figure 1-7: install iturpropag-1.5-py2.py3-non-any.whl using pip command

At the moment the *ITURPropagPY* is installed and you don't need any more those files (downloaded from GitHub) and you can delete them. But it's suggested to keep the content of "*Examples*" folder which is useful to have some issue how to use the software.

### 1.2.2 Install from pip stream

Not Available Now!

Since the "ESA-PL v2.3" license is not supported by <https://pypi.org>

## 2. Define Python Interpreter

After following the all steps in chapter 1, it's the time to define the python interpreter for VSCode.

- Normally, After creating an empty \*.py file or opening any python script file, the VSCode automatically distinguish the file type and it's appropriate compiler installed in your computer. To understand if it does correctly in your computer, look at the **Block 3** in Figure 2-1 to see if the python interpreter is distinguished or not. For example, here the *Python 3.7.1 64-bit ('base': conda)* is distinguished.
- If VSCode didn't find any python interpreter from your computer automatically or you prefer to choose another python interpreter, push the buttoms "*ctrl+shift+p*" from the keyboard and write below command and enter (Figure 2-1)

```
>python: select interpreter
```

- From the opened listed choose any of the python software that is installed in your computer. For example as in Figure 2-2, in my computer two different python are installed and I choose the "*Python 3.7.3 64-bit*"

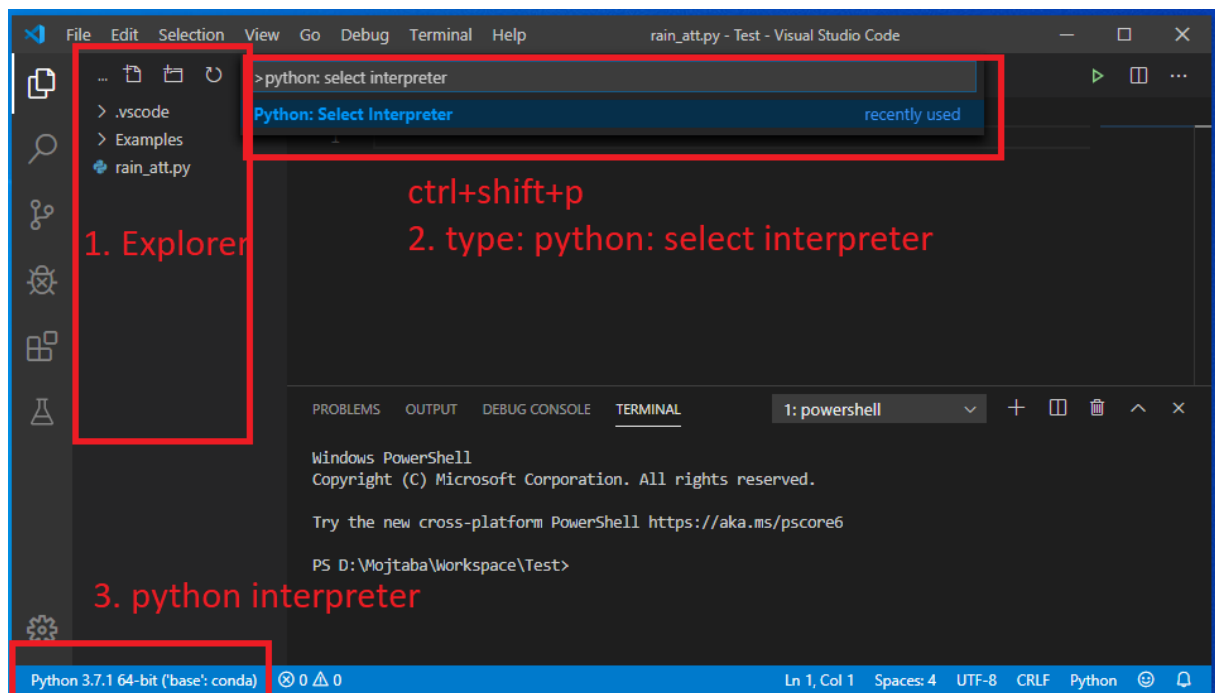


Figure 2-1: VSCode setting for python interpreter

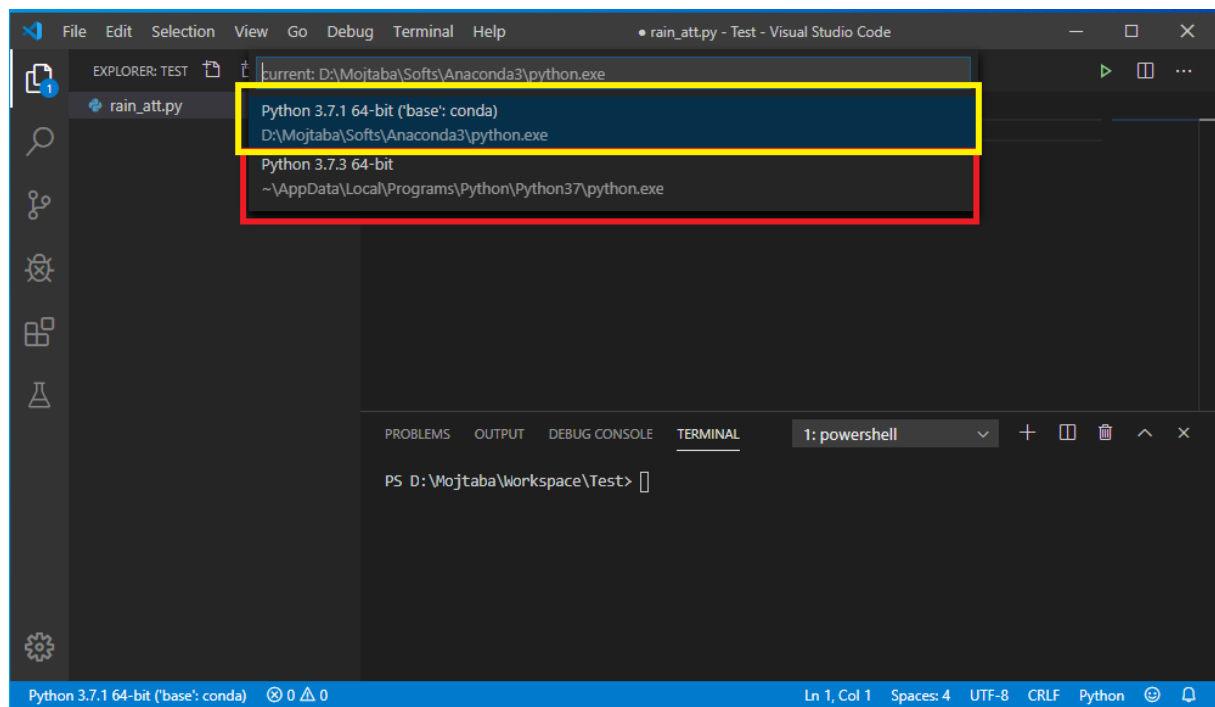


Figure 2-2: Choosing Python Interpreter

### 3. Python Script

To run any command in Python software you can run the command

```
$ python
```

In the terminal and enter (Figure 3-1), and it goes to python software which you can write your command. For exiting from Python space use the command line

```
>>> exit()
```

But for most of the application we have to write a long script. For this reason in the explorer of the VSCode software , do “right click” and create an empty file with the extension of \*.py format.

NOTE: please be sure that the python interpreter distinguished by the VSCode is exactly the one as you wish (Follow the procedure in section 2). If there are a few python installed in your system, you should only choose the one that you have installed the packages for that one !!, for example I have installed all packages only for the “Python 3.7.3 64-bit”. Then for all of my simulation I should choose this python version in my VSCode.

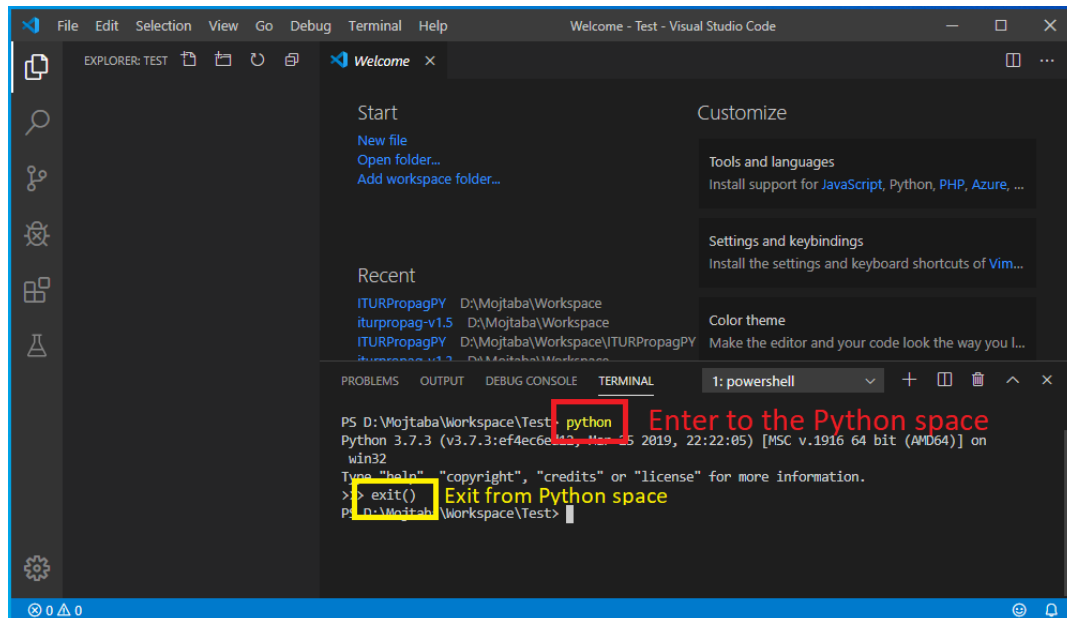


Figure 3-1: Python Space

### 3.1 Atmospheric Attenuation

In this example, we want to calculate the rain, cloud, gaseous and scintillation attenuation for a location with specified link data. It should be noticed that the rain, cloud, gaseous and scintillation attenuation models are in ITU-R P.618, ITU-R P.840, ITU-R P.676 and ITU-R P.618 recommendations, respectively.

#### 3.1.1 Single location vs single probability

To do this first we have to import the desired functions. For this reason, first create an empty file “atm\_att.py” and write the code line:



```
$ from iturpropag.models.iturp618.rain_attenuation import rain_attenuation
$ from iturpropag.models.iturp618.scintillation_attenuation import \
    scintillation_attenuation
$ from iturpropag.models.iturp840.cloud_attenuation import \
    cloud_attenuation
$ from iturpropag.models.iturp676.gaseous_attenuation_slant_path import \
    gaseous_attenuation_slant_path
```

It's nice whenever you are typing the code line in VSCode have a look at the suggestion command introduced by the VSCode software which are so helpful to remember the codes (Figure 3-2)!

**Note:** when you are writing a python script and you want to break a line and keep the continuity of the line use \ and go to the next line.

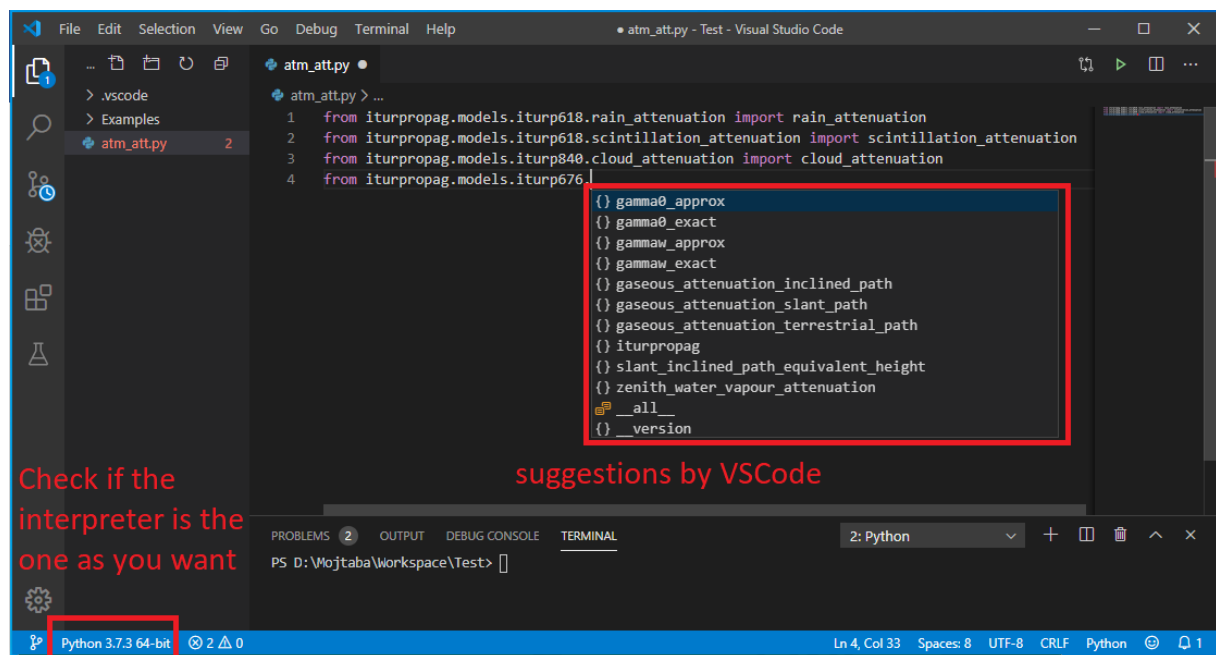


Figure 3-2: VSCode suggestion codes

At the moment you have imported the desired function. But maybe you don't know what information or inputs are needed for this function. To know about this one, keep the "mouse pointer" on the function name and wait the information pop up list appears on the screen. On that screen the necessary information for using that function can be seen (Figure 3-3).

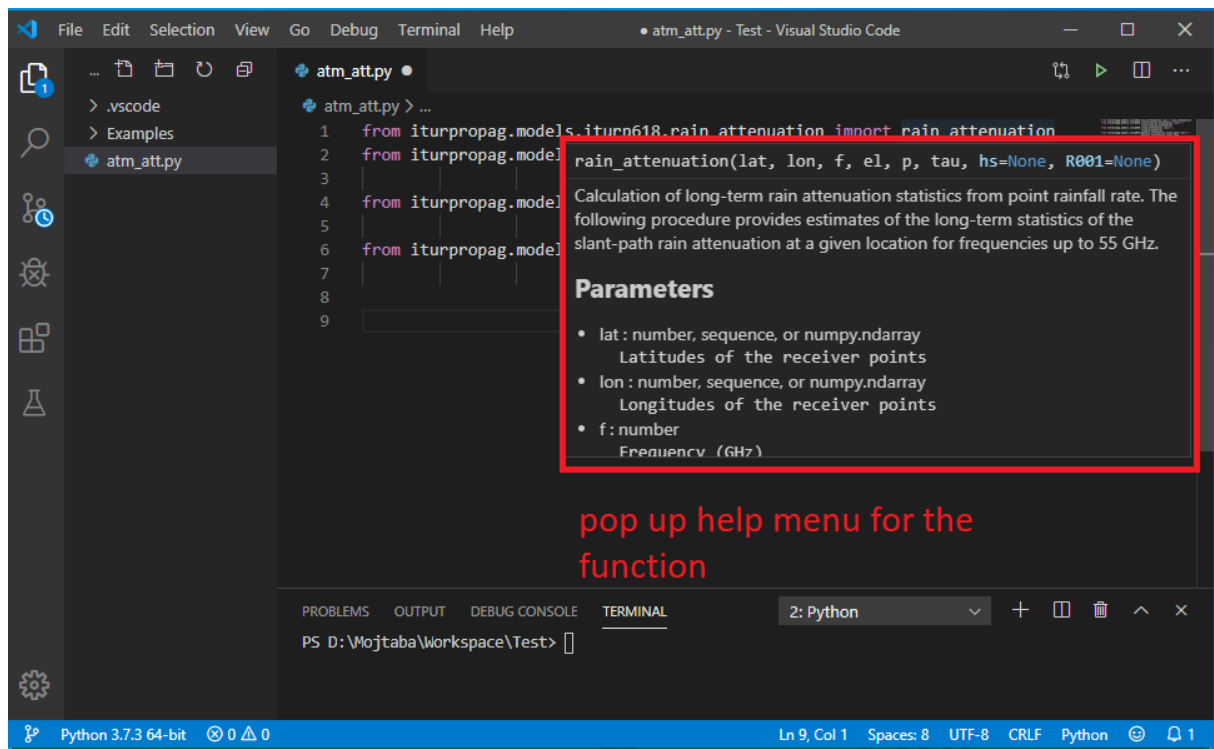


Figure 3-3: pop up help menu

In pop up help menu all the necessary/optional inputs, types and quantity of inputs can be seen. Optional inputs are the ones that are initialized with a value. For example in Figure 3-3 for function `rain_attenuation` the values “`hs=None`” and “`R001=None`” are the optional inputs. In the pop up help menu it is described if these values are left as None value, the function will use other recommendations to get these values. For our first example we will use only necessary inputs.

**Note:** For all of the examples it has tried to use data from the file ‘SG3.xlsx’ in the examples folder to show also the validity of the procedure.

```

$ from iturpropag.models.iturp618.rain_attenuation import rain_attenuation
$ from iturpropag.models.iturp618.scintillation_attenuation import \
    scintillation_attenuation
$ from iturpropag.models.iturp840.cloud_attenuation import \
    cloud_attenuation
$ from iturpropag.models.iturp676.gaseous_attenuation_slant_path import \
    gaseous_attenuation_slant_path
$
$ lat = 51.5
$ lon = -0.14
$ f = 14.25
$ el = 31.0769
$ p = 1
$ tau = 0
    $ D = 1
    $ eta = 0.6
    $ T = 283.61
    $ P = 1004.96 + 17.83
    $ rho = 13.629
$
$ Ar = rain_attenuation(lat, lon, f, el, p, tau)
    $ As = scintillation_attenuation(lat, lon, f, el, p, D, eta)
    $ Ac = cloud_attenuation(lat, lon, el, f, p)
    $ Ag = gaseous_attenuation(f, el, rho, P, T)
$
$ print(Ar, As, Ac, Ag)

```

For gaseous attenuation we need the total pressure as one of the input, in above codes you can see for total pressure, ( $P = 1004.96 + 17.83$ ) dry pressure and water vapour pressure are added together to create total pressure. To run the code: (a) “right click” on the script and click on the “Run Python File in the Terminal” or (b) click the green triangular on upper right side of the script (Figure 3-4)

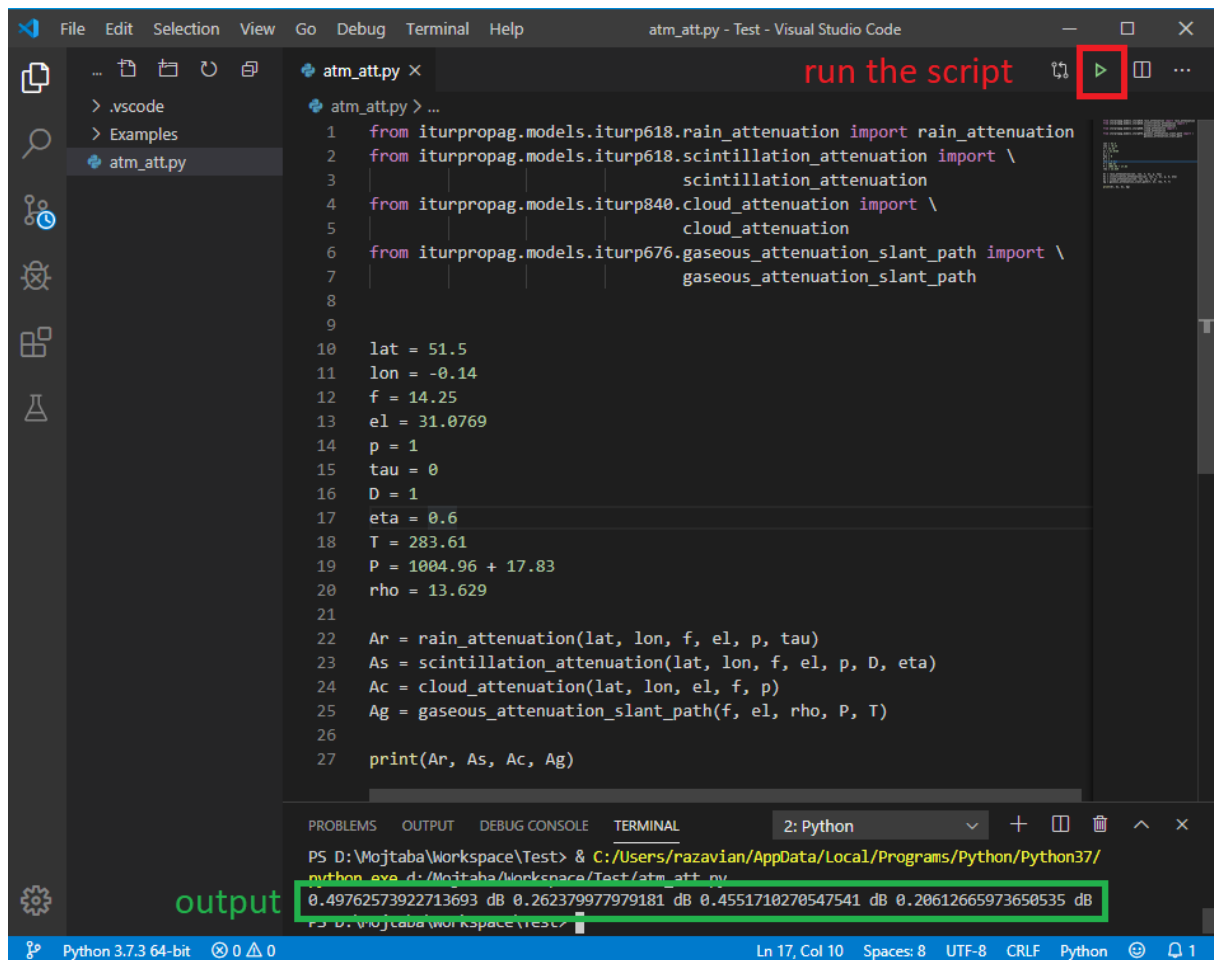


Figure 3-4: Run the Script

### 3.1.2 Single location vs different probability

Now for the second example it's assumed to calculate the atmospheric attenuation for different " $p$ " (probability). For this reason, the probability should be written as an array. The useful package for working with arrays and other special functions is "*numpy*". Then, follow next code lines:

```

1$ import numpy as np
2$ from iturpropag.models.iturp618.rain_attenuation import rain_attenuation
3$ from iturpropag.models.iturp618.scintillation_attenuation import \
4$     scintillation_attenuation
5$ from iturpropag.models.iturp840.cloud_attenuation import \
6$     cloud_attenuation
7$ from iturpropag.models.iturp676.gaseous_attenuation_slant_path import \
8$     gaseous_attenuation_slant_path
9$
10$ lat = 51.5
11$ lon = -0.14
12$ f = 14.25
13$ el = 31.0769
14$ p = np.array([0.001, 0.01, 0.1, 1])
15$ tau = 0
16$ D = 1
17$ eta = 0.6
18$ T = 283.61
19$ P = 1004.96 + 17.83
20$ rho = 13.629
21$
22$ Ar = rain_attenuation(lat, lon, f, el, p, tau)
23$ As = scintillation_attenuation(lat, lon, f, el, p, D, eta)
24$ Ac = cloud_attenuation(lat, lon, el, f, p)
25$ Ag = gaseous_attenuation(f, el, rho, P, T)
26$
27$ print('Ar=', Ar, '\nAs=', As, '\nAc=', Ac, '\nAg=', Ag)

```

For simplicity when a package is imported into python the name of the package can be change into simpler name for example in above codes the name of the package “*numpy*” is changed to “*np*” in line 1\$ for simpler usage. To create an array using numpy package use this package as in line 14\$ in above code lines.

```
1 import numpy as np
2 from iturpropag.models.iturp618.rain_attenuation import rain_attenuation
3 from iturpropag.models.iturp618.scintillation_attenuation import \
4     scintillation_attenuation
5 from iturpropag.models.iturp840.cloud_attenuation import \
6     cloud_attenuation
7 from iturpropag.models.iturp676.gaseous_attenuation_slant_path import \
8     gaseous_attenuation_slant_path
9
10
11 lat = 51.5
12 lon = -0.14
13 f = 14.25
14 el = 51.0709
15 p = np.array([0.001, 0.01, 0.1, 1])
16 tau = 0
17 D = 1
18 eta = 0.6
19 T = 283.61
20 P = 1004.96 + 17.83
21 rho = 13.629
22
23 Ar = rain_attenuation(lat, lon, f, el, p, tau)
24 As = scintillation_attenuation(lat, lon, f, el, p, D, eta)
25 Ac = cloud_attenuation(lat, lon, el, f, p)
26 Ag = gaseous_attenuation_slant_path(f, el, rho, P, T)
27
28 print('Ar=', Ar, '\nAs=', As, '\nAc=', Ac, '\nAg=', Ag)
```

use numpy to create an array

```
PS D:\Mojtaba\Workspace\Test> & C:/Users/razavian/AppData/Local/Programs/Python/Python37/
python.exe d:/Mojtaba/Workspace/Test/atm_att.py
Ar= [14.9513575  6.82430841  2.19515933  0.49762574] dB
As= [0.91177042 0.62936211 0.42356874 0.26237998] dB
Ac= [1.09293271 0.88935238 0.68577205 0.45517103] dB
Ag= 0.20612665973650535 dB
PS D:\Mojtaba\Workspace\Test>
```

output

Figure 3-5: Calculate the atmospheric attenuation for different probability

### 3.1.3 Single location vs different probability and plotting output

To enter more points in variable “p” and plot the output it’s better to use the “*numpy.arange*” function to create a big array for variable “p” and use the “*matplotlib*” to plot the output. For this reason write the following code lines:

```

1$ import numpy as np
2$ import matplotlib.pyplot as plt
3$ from iturpropag.models.iturp618.rain_attenuation import rain_attenuation
4$ from iturpropag.models.iturp618.scintillation_attenuation import \
5$     scintillation_attenuation
6$ from iturpropag.models.iturp840.cloud_attenuation import \
7$     cloud_attenuation
8$ from iturpropag.models.iturp676.gaseous_attenuation_slant_path import \
9$     gaseous_attenuation_slant_path
10$
11$ lat = 51.5
12$ lon = -0.14
13$ f = 14.25
14$ el = 31.0769
15$ p = np.arange(0.001, 10, 0.01)
16$ tau = 0
17$ D = 1
18$ eta = 0.6
19$ T = 283.61
20$ P = 1004.96 + 17.83
21$ rho = 13.629
22$
23$ Ar = rain_attenuation(lat, lon, f, el, p, tau)
24$ As = scintillation_attenuation(lat, lon, f, el, p, D, eta)
25$ Ac = cloud_attenuation(lat, lon, el, f, p)
26$ Ag = gaseous_attenuation(f, el, rho, P, T)
27$
28$ plt.figure()
29$ plt.plot(p, Ar.value, label='rain attenuation')
30$ plt.plot(p, As.value, label='scintillation attenuation')
31$ plt.plot(p, Ac.value, label='cloud attenuation')
32$
33$ plt.legend()
34$ plt.xscale('log')
35$ plt.xlabel('Percentage of the time [%]')
36$ plt.ylabel('Attenuation [dB]')
37$ plt.show()

```

In line 15\$ the function “*numpy.arange(start, stop, step)*” is used to create an array starting from 0.001 and stop in 10 with the 0.01 discrimination. The lines 28\$-31\$ are used to create a figure and plot the rain, scintillation and cloud attenuation vs probability, line 34\$ change the x-axis to logarithmic scale. Be sure that line 37\$ is necessary to see the figure, if this line is skipped the figure will not be shown. (Figure 3-6).

**Note:** most of the outputs of the functions in ITURPropagPY package are quantity. It means it has a value and a unit. For example in line 29\$-31\$ “*Ar.value*”, “*As.value*” and “*Ac.value*” are used to access only the values of the variable. The gaseous attenuation is not dependent on percentage of the time.

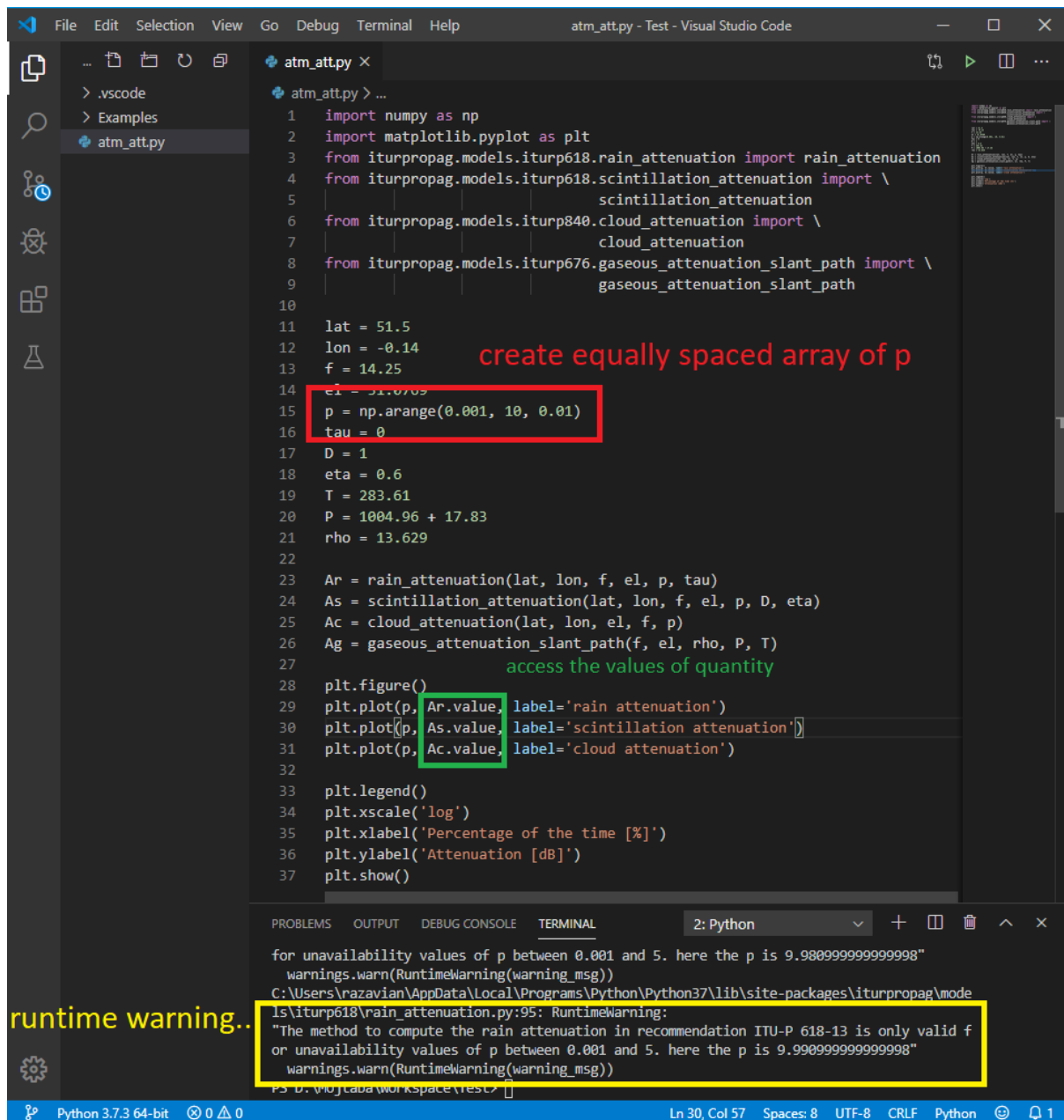


Figure 3-6: plot atmospheric attenuation vs probability

**NOTE:** When this code is run, you will see many “warnings” printed inside the terminal, which say that the method in ITU-R P.618 is only valid for probability between 0.001 and 5 which the probability used here are more than 5%. However, the model still runs for those probabilities more than 5% and this warning is only used to make you be aware of the limitation of this model. For any reason, if you don’t like to see these warnings and you completely know these limitations and want to ignore these messages, by writing a simple code in your script these warnings will be disabled:

```

$ import warnings
$ warnings.filterwarnings("ignore")

```



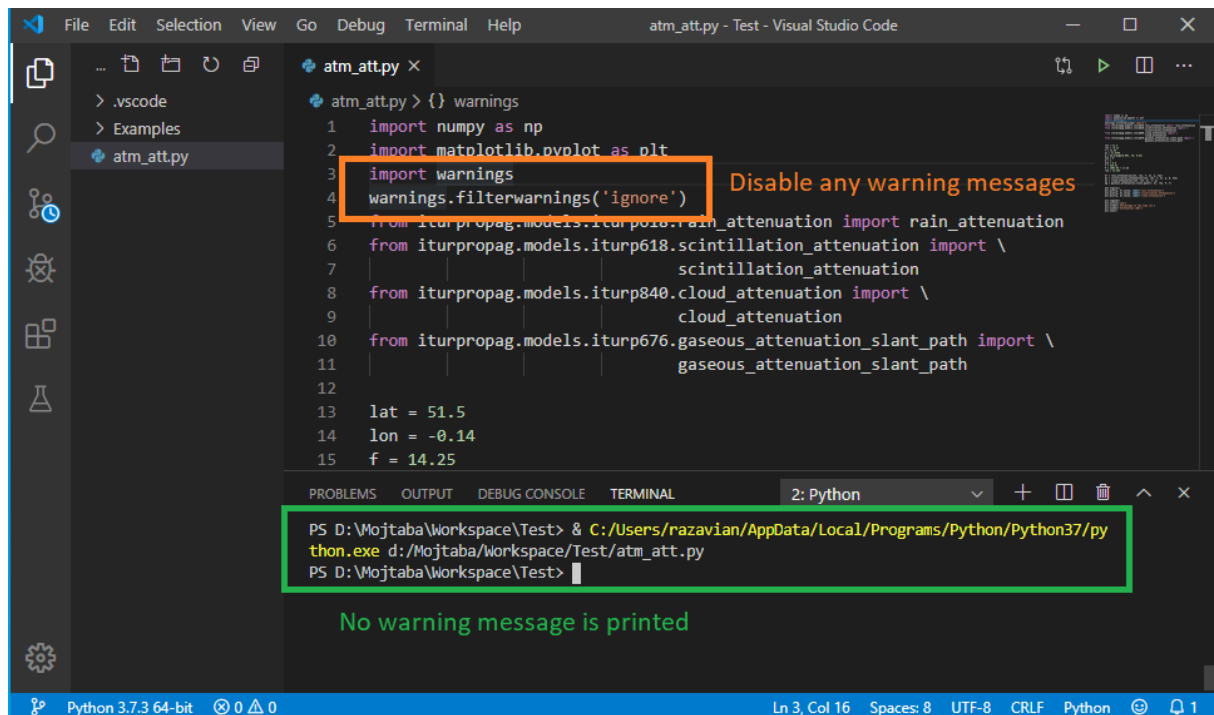


Figure 3-7: Disabling the warning messages

The output of the run will be the Figure 3-8.

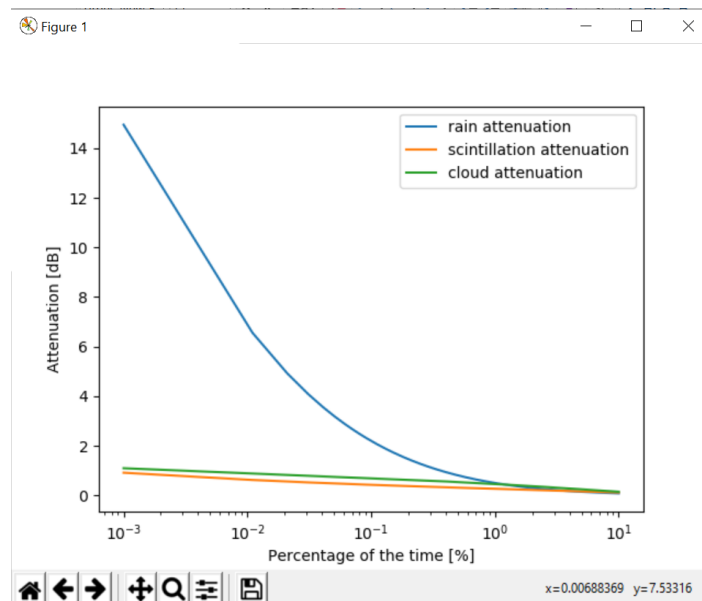


Figure 3-8: rain attenuation vs probability

### 3.2 Export data

To export data into different formats like txt, csv, excel and h5, I strongly suggest to use “*pandas*” python library, which is the best for handling huge data in python (<https://pandas.pydata.org/pandas-docs/stable/>). In Figure 1-4 you can find the package of “*pandas*” is added to be installed like other packages.

### 3.2.1 Export data to CSV file:

For exporting, any data to csv file please follow next code lines:

```
1$ import numpy as np
2$ import pandas as pd
3$ import warnings
4$ warnings.filterwarnings('ignore')
5$ from iturpropag.models.iturp618.rain_attenuation import rain_attenuation
6$
7$ lat = 51.5
8$ lon = -0.14
9$ f = 14.25
10$ el = 31.0769
11$ p = np.arange(0.001, 10, 0.001)
12$ tau = 0
13$
14$ Ar = rain_attenuation(lat, lon, f, el, p, tau)
15$
16$ data = {'Rain Attenuation [dB]' : Ar,
17$         'probability [%]' : p}
18$ df = pd.DataFrame(data)
19$ df.to_csv('rain_att.csv', index=False)
```

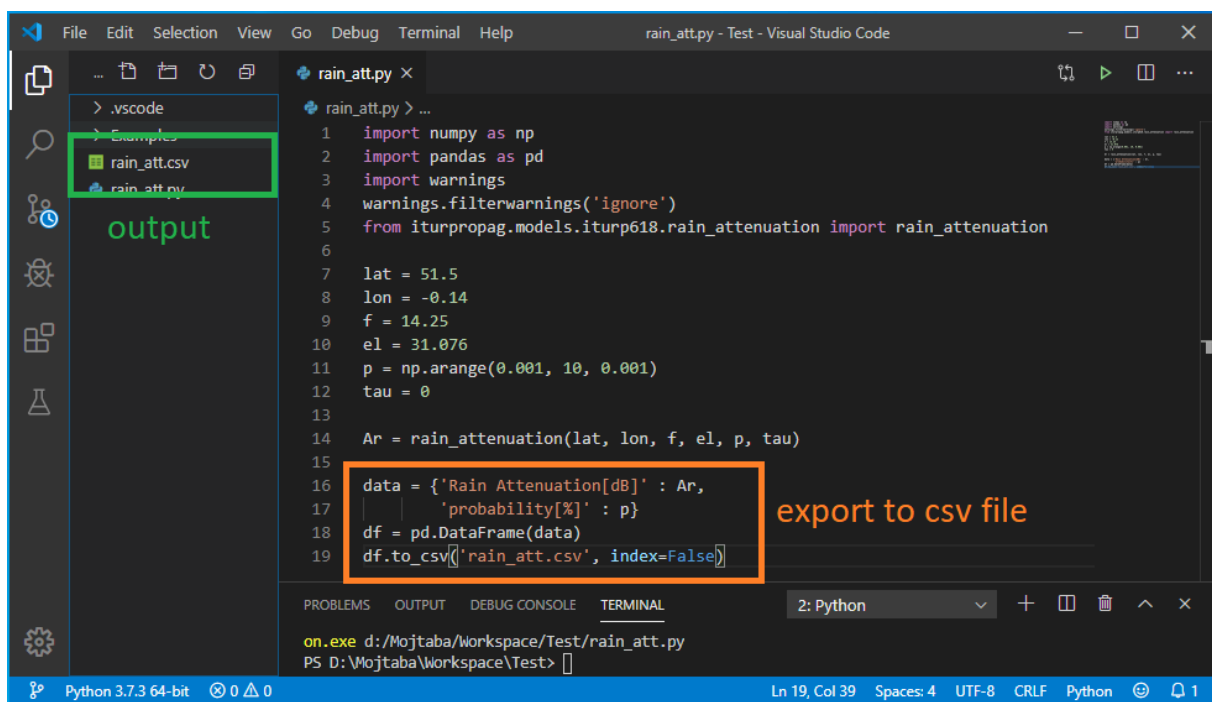
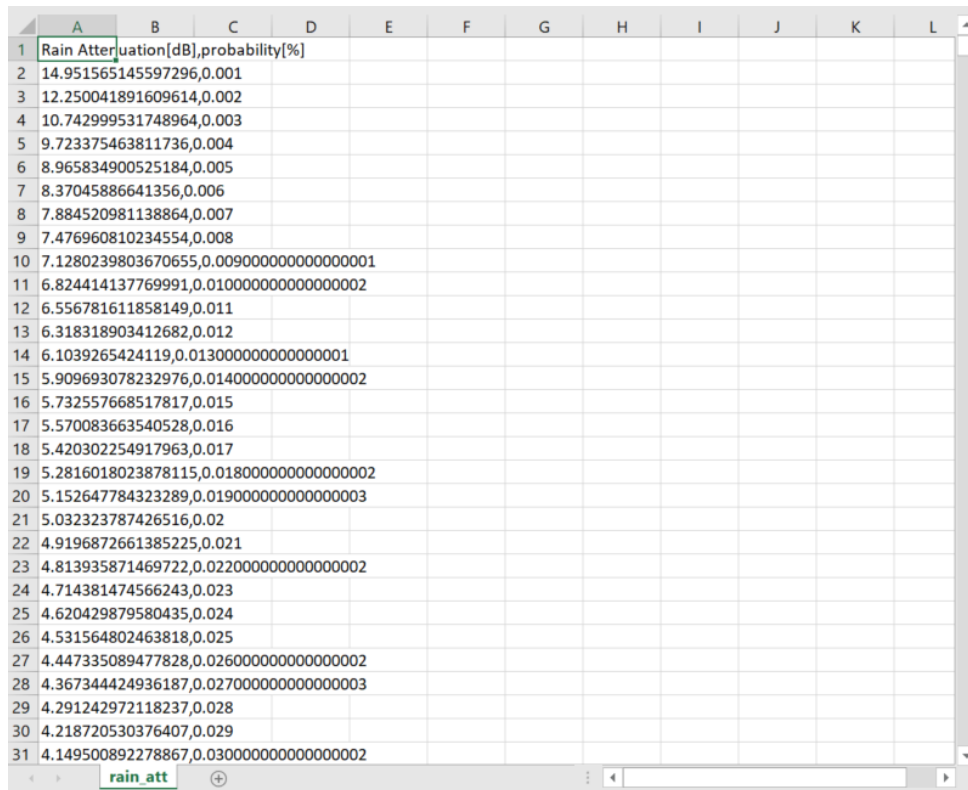


Figure 3-9: Export data to CSV file

The line 2\$ will import “pandas” and change the name to “pd” for simpler using. The lines 16\$-17\$ will create a dictionary variable, which any name written inside ‘ ’ will be the header for that variable inside the exported file. Line 18\$ will create a Data Frame variable from the original data. Finally line 19\$ is used to export the previous Data Frame variable into \*.csv file with the name of ‘rain\_att.csv’. Be noted that the option “index=False” is used to prevent of printing the index number of the columns into the exported file. Figure 3-10 shows the content of the exported CSV file.

The full documentation of exporting the data to csv file can be found at the “*pandas*” website : [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/io.html#io-store-in-csv](https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html#io-store-in-csv)



	A	B	C	D	E	F	G	H	I	J	K	L
1	Rain Attenuation[dB],probability[%]											
2	14.951565145597296,0.001											
3	12.250041891609614,0.002											
4	10.742999531748964,0.003											
5	9.723375463811736,0.004											
6	8.965834900525184,0.005											
7	8.37045886641356,0.006											
8	7.884520981138864,0.007											
9	7.476960810234554,0.008											
10	7.1280239803670655,0.009000000000000001											
11	6.824414137769991,0.010000000000000002											
12	6.556781611858149,0.011											
13	6.318318903412682,0.012											
14	6.1039265424119,0.013000000000000001											
15	5.909693078232976,0.014000000000000002											
16	5.732557668517817,0.015											
17	5.570083663540528,0.016											
18	5.420302254917963,0.017											
19	5.2816018023878115,0.018000000000000002											
20	5.152647784323289,0.019000000000000003											
21	5.032323787426516,0.02											
22	4.9196872661385225,0.021											
23	4.813935871469722,0.022000000000000002											
24	4.714381474566243,0.023											
25	4.620429879580435,0.024											
26	4.531564802463818,0.025											
27	4.447335089477828,0.026000000000000002											
28	4.367344424936187,0.027000000000000003											
29	4.291242972118237,0.028											
30	4.218720530376407,0.029											
31	4.149500892278867,0.030000000000000002											

Figure 3-10: Exported CSV file

### 3.2.2 Export data to text file:

For exporting the data into text format, the same command of section 3.2.1 can be used but with a little bit change at line 19\$

```
16$ data = {'Rain Attenuation [dB]' : Ar,
17$         'probability [%]' : p}
18$ df = pd.DataFrame(data)
19$ df.to_csv('rain_att.txt', index=False, sep='t')
```

In line 19\$ the option for separation the values is added. Here “sep='t'” is used which means the columns values are separated with a tab. The output will be look like the Figure 3-11

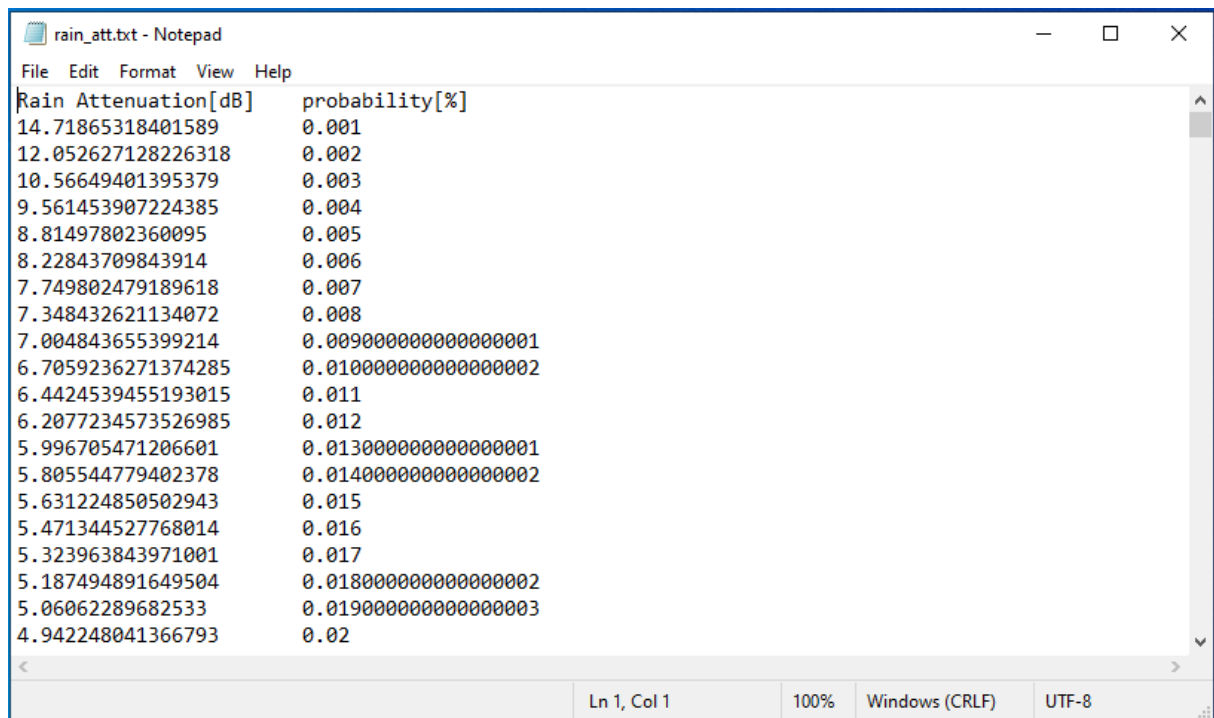


Figure 3-11: Exported text file

### 3.2.3 Export data to Excel file

For exporting the data into Excel file, the python libraries “*lxml*” and “*xlsxwriter*” should be installed.

```
$ pip install lxml xlsxwriter
```

Then use the command lines as in section 3.2.1 with the changes in lines 19\$-21\$ (Figure 3-12):

```
16$ data = {'Rain Attenuation [dB]' : Ar,
17$         'probability [%]' : p}
18$ df = pd.DataFrame(data)
19$ writer = pd.ExcelWriter('rain_att.xlsx')
20$ df.to_excel(writer, sheet_name='rain attenuation', index=False)
21$ writer.save()
```

The full documentation for using this library can be found at pandas website :

[https://pandas.pydata.org/pandas-docs/stable/user\\_guide/io.html#io-excel-writer](https://pandas.pydata.org/pandas-docs/stable/user_guide/io.html#io-excel-writer)

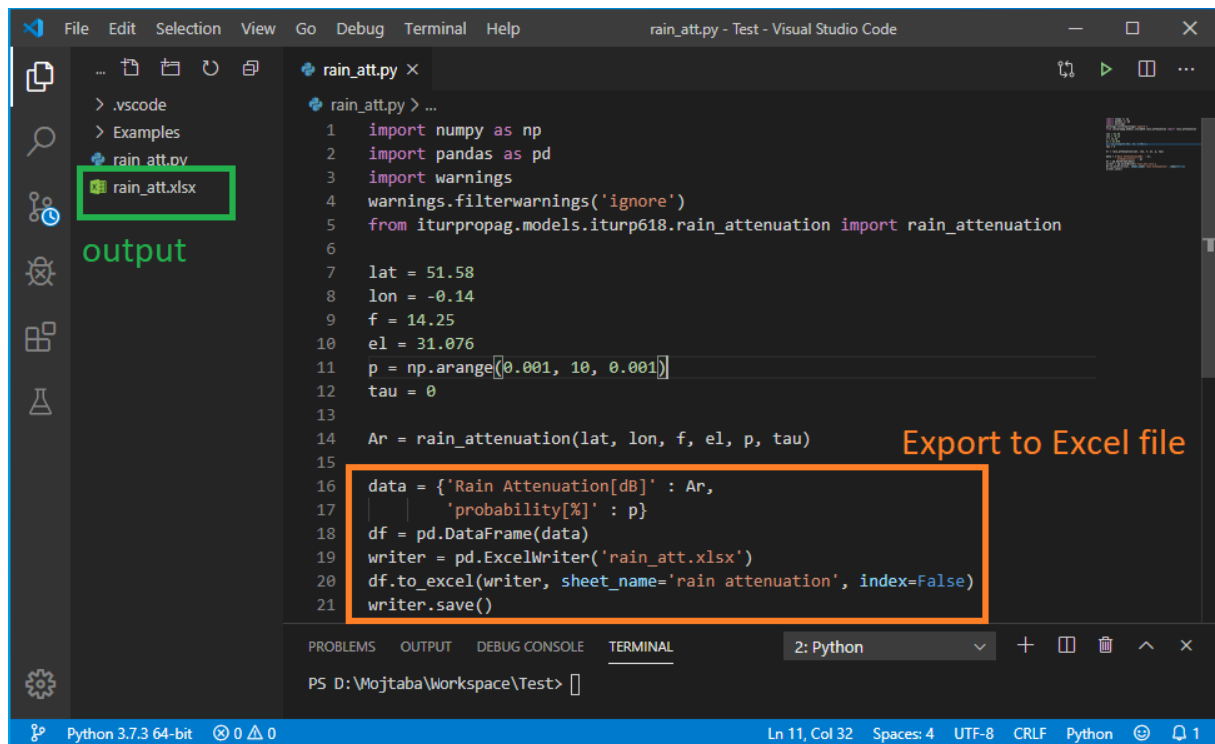


Figure 3-12: Export to Excel file

### 3.2.4 Export data to h5 file:

For exporting the data into h5 format the python library “tables” should be installed:

```
$ pip install tables
```

Then use the command lines as in section 3.2.1 with the changes in lines 19\$

```

16$ data = {'Rain Attenuation [dB]' : Ar,
17$         'probability [%]' : p}
18$ df = pd.DataFrame(data)
19$ df.to_hdf('rain_att.h5', key='rain_att', mode='w', complevel=9)

```

In line 19\$ the option “key=” is used to name the data in the h5 file and “mode=’w’” means we want to write a new file. The option “complevel=9” will compress the output data with the specified level in range 0-9 with maximum compression at 9 and minimum compression at 0. This option is so useful for large files. The full documentation about this library can be read at pandas website:

[https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.to\\_hdf.html](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.to_hdf.html)

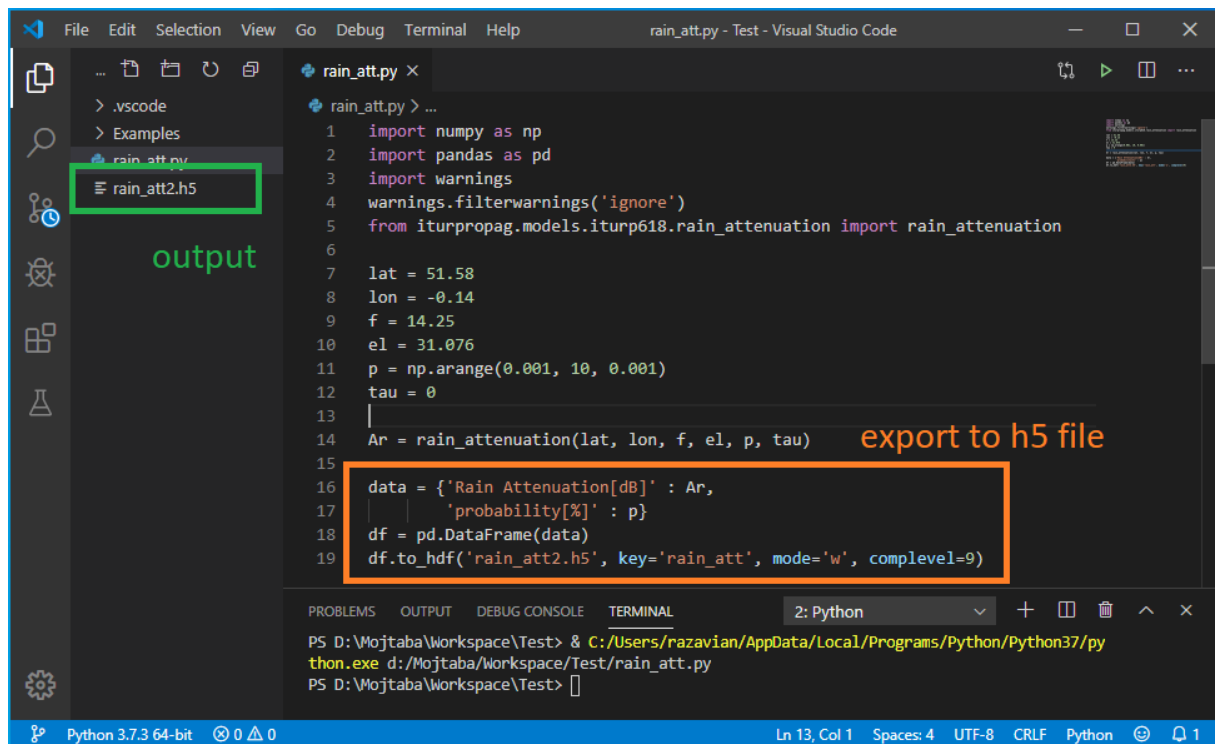


Figure 3-13: export data to h5 file