

SigMT - User Manual

v. 1

K.S. Ajithabh

June 11, 2022

Introduction

SigMT is a python package designed for the processing of the raw magnetotelluric (MT) data to obtain the MT impedance and tipper estimates. It works in an automated way, so that manual time series inspection and editing are not required. Mahalanobis based data selection tool is implemented in the package to avoid the manual editing of time series. The final impedance estimation is done using the robust estimation method. Different data selection tools such as coherency threshold, polarization direction are included in this package. This document is a guide to use the SigMT package. At present, only ‘.ats’ file formats from ADU07 (Metronix Geophysics) is supported.

Contents

1	Supported file formats	3
2	Getting started	3
3	Overview of the package	5
4	An example	7
4.1	Running the program	7
4.2	Creating EDI file	10
4.3	Data selection tools	15
4.4	Remote reference	17

1 Supported file formats

- *.ats file format from ADU-07 by Metronix Geophysics

2 Getting started

The program is written as a project in Spyder IDE with the support of ‘Anaconda’ Python Distribution. I suggest installation of the Anaconda first to start with this package. You can download Anaconda for Windows/Linux/Mac from Anaconda website. Please install Anaconda3-2021.11 version from the Anaconda archives using the link <https://repo.anaconda.com/archive/>. Because newer version is showing some errors. Anaconda provides all necessary python modules for the scientific computations.

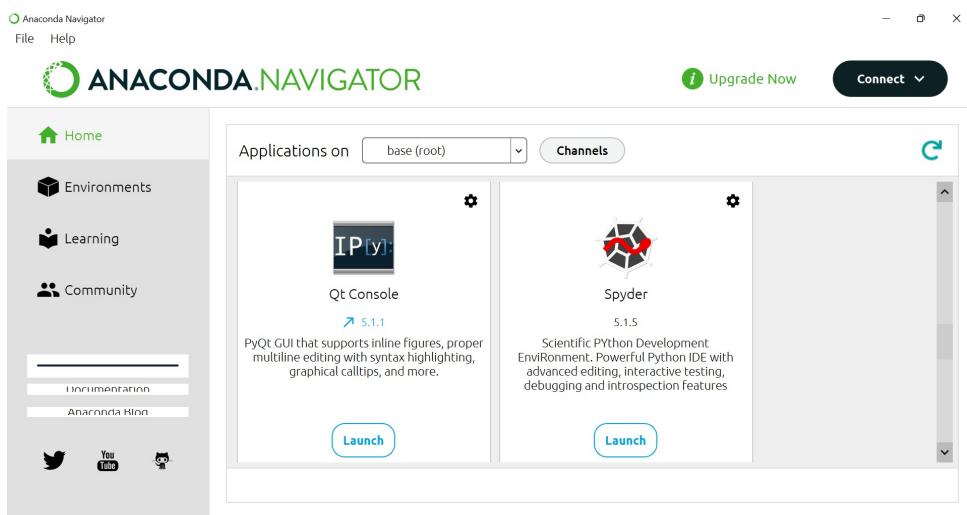


Figure 1: Anaconda Navigator

After installation of Anaconda, you can check in Anaconda Navigator (Figure 1) whether Spyder is installed or not. If not installed by default, please install Spyder using install button in Anaconda Navigator.

Once Anaconda and Spyder is ready, you can download the SigMT package as zip from GitHub (Figure 2). Extract the compressed folder to your local drive. Then open ‘Spyder IDE’, select Projects and click ‘Open Project’ (Figure 3).

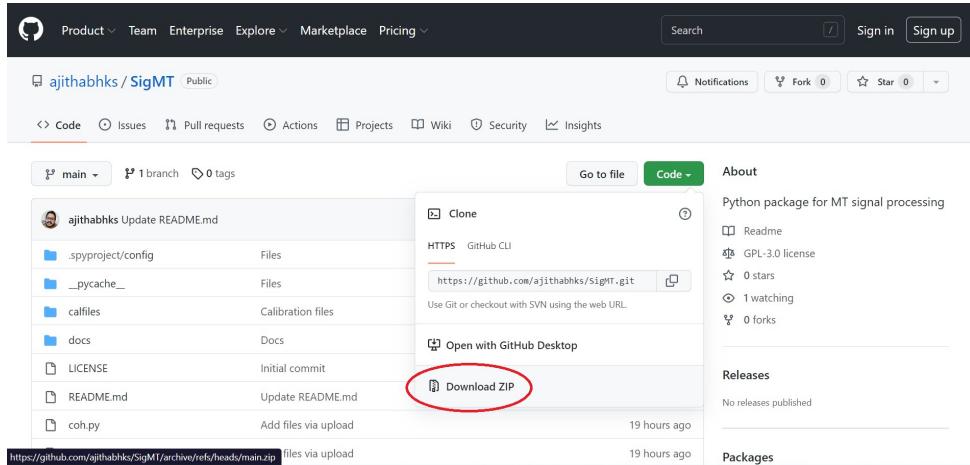


Figure 2: Download the package

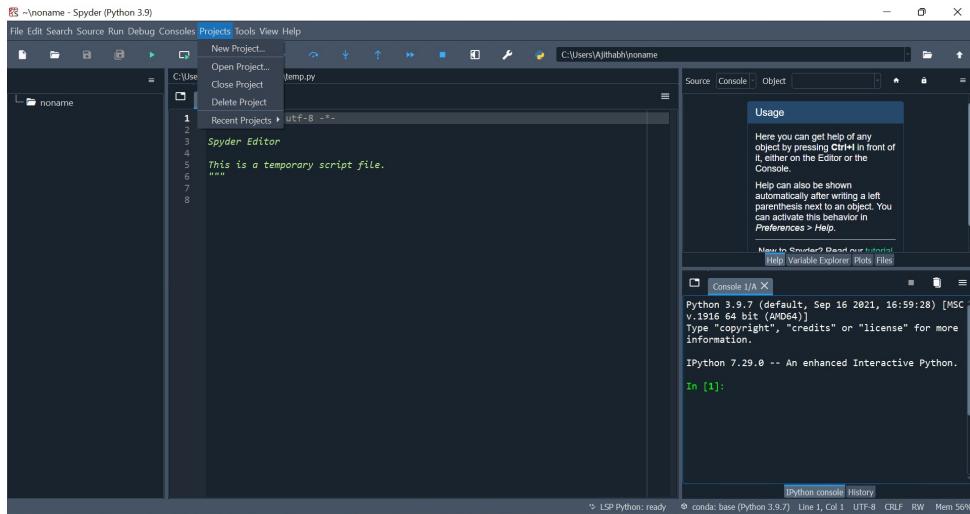


Figure 3: Open Project in Spyder

Navigate to extracted SigMT project folder and select it. Now you are in SigMT package. Before running the package, you have to install the ‘mne’ package using the pip. Run following command in Spyder console to install it.

```
pip install mne
```

Now add the calibration files of your MFS06 sensors to the ‘calfiles’ folder in SigMT package folder. Please use sensor number as file name (as in Figure 4), otherwise program will not read it. Some sample files are given by default in ‘calfiles’ folder. Please try to make your files similar to it.

Name	Date modified	Type	Size
250.TXT	11-06-2012 18:06	Text Document	5 KB
251.TXT	11-06-2012 18:06	Text Document	5 KB
252.TXT	11-06-2012 18:06	Text Document	5 KB
253.TXT	11-06-2012 18:06	Text Document	5 KB
254.TXT	11-06-2012 18:06	Text Document	5 KB
255.TXT	11-06-2012 18:07	Text Document	5 KB
300.TXT	27-04-2020 01:01	Text Document	5 KB
301.TXT	27-04-2020 01:02	Text Document	5 KB
302.TXT	27-04-2020 01:02	Text Document	5 KB
303.TXT	27-04-2020 01:01	Text Document	5 KB
311.TXT	27-04-2020 01:02	Text Document	5 KB
314.TXT	27-04-2020 01:02	Text Document	5 KB
315.TXT	27-04-2020 01:02	Text Document	5 KB
316.TXT	27-04-2020 01:02	Text Document	5 KB
318.TXT	27-04-2020 01:03	Text Document	5 KB
561.txt	19-08-2020 15:57	Text Document	5 KB

Figure 4: Calibration files

Now you are all set to start processing your MT data!!

3 Overview of the package

The file ‘main_script.py’ is the file you need to run the package for single site processing. But, if you need to carry out Remote reference, please use ‘main_script_RR.py’ file.

The files such as mtproc.py, mtprocRR.py, coh.py, coherency.py, mahaDist.py, self-stack.py, tipper.py are written to perform different tasks in package and need not to be edited.

I suggest editing of main_script.py and main_script_RR.py as per your needs. The more description of these files are given below.

First of all, create a project folder and copy all your MT sites in it (as in Figure 5).

This PC > DATA (D:) > NGRI > FIELD RAW DATA > demo			
Name	Date modified	Type	Size
KL33A	27-04-2022 16:06	File folder	
LU16	27-04-2022 16:06	File folder	
LU33A	27-04-2022 16:06	File folder	

Figure 5: Project folder

The measurements should be in the site folder as in Figure 6. There should not be any other extra folders inside the site folder.

Name	Date modified	Type	Size
meas_2019-08-29_09-01-00	27-04-2022 16:05	File folder	
meas_2019-08-29_09-04-00	27-04-2022 16:05	File folder	
meas_2019-08-29_09-11-00	27-04-2022 16:05	File folder	
meas_2019-08-29_09-18-00	27-04-2022 16:05	File folder	
meas_2019-08-29_09-25-00	27-04-2022 16:05	File folder	
meas_2019-08-29_14-30-00	27-04-2022 16:05	File folder	
meas_2019-08-29_14-33-00	27-04-2022 16:05	File folder	
meas_2019-08-29_14-36-00	27-04-2022 16:05	File folder	
meas_2019-08-29_14-39-00	27-04-2022 16:05	File folder	
meas_2019-08-29_15-01-00	27-04-2022 16:05	File folder	
meas_2019-08-29_15-23-00	27-04-2022 16:06	File folder	
meas_2019-08-29_16-05-00	27-04-2022 16:06	File folder	
meas_2019-08-29_16-27-00	27-04-2022 16:06	File folder	
meas_2019-08-29_16-49-00	27-04-2022 16:06	File folder	
meas_2019-08-29_16-52-00	27-04-2022 16:06	File folder	
meas_2019-08-29_21-54-00	27-04-2022 16:06	File folder	

Figure 6: Site folder

There are three places, code can be modified in ‘main_script.py’ and ‘main_script_RR.py’ files to provide site folder path, decimation and data selection constraints.

First edit is required to provide the sites folder path. Copy the location of folder where sites are kept and copy to the variable ‘project_path’ as in Figure 7.

```

D:/Pyth/demo - Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:/Pyth/demo/main_script.py
D:/Pyth/demo/main_script.py
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon May  4 16:49:35 2020
4
5 @author: AJITHABH
6
7 import mtproc, coh, coherence, tipper, mahaDist
8 from scipy import signal
9 from matplotlib import pyplot as plt
10 import numpy as np
11 import os
12 import time
13 import math
14 #
15 #
16 # provide project path where sites are kept
17 #
18 project_path = 'D:/NGRI/FIELD RAW DATA/demo/' # Red circle highlights this line
19 #
20 #
21 os.chdir(project_path)
22 sites = [d for d in os.listdir('.') if os.path.isdir(d)]
23 # all site names are stored in variable sites
24 # ask user for site name on the project direc.
25 print(sites)
26 selectedsite = input("Enter the site: ")
27 siteindex = sites.index(selectedsite)
28 measid = mtproc.measid(siteindex)
29 del siteindex
30 os.chdir(project_path+selectedsite)
31 all_meas = [d for d in os.listdir('.') if os.path.isdir(d)]
32 for i in range(len(all_meas)):
33     for j in

```

Figure 7: Give project path (sites folder)

Second edit is required in case you require decimation. The highlighted portions in Figure 8 is written to perform decimation. Keep ‘dflag’ 1 if you need to used decimation, else always keep 0. Suppose you need to decimate the data sampled at 32 Hz. If you use [8,4], then the data will be first decimated to $32/8 = 4$ Hz, then $4/4 = 1$ Hz. So, the output will be 1 Hz. Direct decimation using [32] is not recommended. If you used [8,8,4], the data will be decimated to 0.125 Hz. Similarly, you can decimate to any sampling frequencies.

```

43 timer_start = time.time()
44 #
45 ##### Time series reading starts #####
46 #
47 # procinfo = {}
48 # [ts,procinfo['fs'],procinfo['sensor_no'],timeline,procinfo['ChoppStat'],loc] = mtproc.t
49 # trend removed
50 # n = len(km)
51 # H = np.zeros((n,n))
52 # H = np.eye(n)
53 # H = np.ones((n,n))
54 #
55 dflag = 1
56 if dflag == 1:
57     decimate = [8,4]
58     for d in decimate:
59         ts[tsEx] = signal.decimate(ts.get('tsEx'), d, n=None, ftype='fir')
60         ts[tsEy] = signal.decimate(ts.get('tsEy'), d, n=None, ftype='fir')
61         ts[tsHx] = signal.decimate(ts.get('tsHx'), d, n=None, ftype='fir')
62         ts[tsHy] = signal.decimate(ts.get('tsHy'), d, n=None, ftype='fir')
63         ts[tsHz] = signal.decimate(ts.get('tsHz'), d, n=None, ftype='fir')
64         procinfo['fs'] = procinfo.get('fs')/d
65         procinfo['nofs'] = len(ts[tsEx])
66         procinfo['notch'] = 0 # Notch flag 1 - On, 0 - Off
67     print('-----')
68     print('Measurement directory: ' + all_meas[select_meas])
69     print('Fs = ' + str(procinfo.get('fs')) + ' Hz')
70     print('Sensor numbers: ')
71     print(procinfo.get('sensor_no'))
72     print('Length of time series = ' + str(procinfo.get('nofs')))
73     print('-----')
74     print('-----')
75     print('-----')
76     print('-----')
77     print('-----')
78     print('-----')
79     print('-----')
80     print('-----')
81     print('-----')
82     print('-----')
83     print('-----')
84     print('-----')
85     print('-----')
86     print('-----')
87     print('-----')
88     print('-----')
89     print('-----')
90     print('-----')
91     print('-----')
92     print('-----')
93     print('-----')
94     print('-----')
95     print('-----')
96     print('-----')
97     print('-----')
98     print('-----')
99     print('-----')
100    print('-----')
101    print('-----')
102    print('-----')
103    print('-----')
104    print('-----')
105    print('-----')
106    print('-----')
107    print('-----')
108    print('-----')
109    print('-----')
110    print('-----')
111    print('-----')
112    print('-----')
113    print('-----')
114    print('-----')
115    print('-----')
116    print('-----')
117    print('-----')
118    print('-----')
119    print('-----')
120    print('-----')
121    print('-----')
122    print('-----')
123    print('-----')
124    print('-----')
125    print('-----')
126    print('-----')
127    print('-----')
128    print('-----')
129    print('-----')
130    print('-----')
131    print('-----')
132    print('-----')
133    print('-----')
134    print('-----')
135    print('-----')
136    print('-----')
137    print('-----')
138    print('-----')
139    print('-----')
140    print('-----')
141    print('-----')
142    print('-----')
143    print('-----')
144    print('-----')
145    print('-----')
146    print('-----')
147    print('-----')
148    print('-----')
149    print('-----')
150    print('-----')
151    print('-----')
152    print('-----')
153    print('-----')
154    print('-----')
155    print('-----')

```

Figure 8: Decimation

Third edit is required to control data selection tools. The highlighted portions in Figure 9 is written to perform data selection. Coherency threshold and polarization direction based selections are included in the package. More detailed description is given in section 4.3.

```

124 ctflag == 0
125 if ctflag == 1:
126     CcohThre = [0.6,0.6,0.6,0.6,0.6,0.6,0.6,0.6,0.6,0.6,0.6,0.6]
127     for i in range(np.shape(AllcohEx)[0]):
128         for j in range(np.shape(AllcohEx)[1]):
129             if AllcohEx[i,j] < CcohThre[i]:
130                 cohMatrixEx[i,j] = 0
131             else:
132                 cohMatrixEx[i,j] = 1
133             if AllcohExY[i,j] < CcohThre[i]:
134                 cohMatrixExY[i,j] = 0
135             else:
136                 cohMatrixExY[i,j] = 1
137 #
138 mpdflag = 0
139 if mpdflag == 1:
140     mpdim = [-10,10]
141     for i in range(np.shape(mpdmat)[0]):
142         for j in range(np.shape(mpdmat)[1]):
143             if alpha_deg[i,j] > mpdim[0] and alpha_deg[i,j] < mpdim[1]:
144                 mpdmat[i,j] = 0
145             else:
146                 mpdmat[i,j] = 1
147 #
148 mpdflag = 0
149 if mpdflag == 1:
150     for i in range(np.shape(mpdmat)[0]):
151         for j in range(np.shape(mpdmat)[1]):
152             if j > 360 and j < 408:
153                 mpdmat[i,j] = 0
154             else:
155                 mpdmat[i,j] = 1

```

Figure 9: Data selection tools section

4 An example

4.1 Running the program

I am showing an example of processing here. Give the sites folder path in ‘main_script.py’ file and run the code as in the Figure 10

```

D:\Pyth\demo - Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\Pyth\demo
demo
  -> callfiles
    coh.py
    coherence.py
    mahaDist.py
    main_script.py
    main_script_I
    MPD.py
    mproc.py
    mprocRR.py
    mprocRRR.py
    plotData.py
    plotMD.py
    plotMPD.py
    tipper.py
    tipperR.py
    write2dI_da
    write2dI_he
    write2dI_co
    write2dI_py

```

Run

```

main_script.py X Run
1  # -*- coding: utf-8 -*-
2  """
3  Created on Mon May  4 16:49:35 2020
4
5  @author: AJITHABH
6  """
7  import mproc, coh, coherence, tipper, mahaDist
8  from scipy import signal
9  from matplotlib import pyplot as plt
10 import numpy as np
11 import os
12 import time
13 import math
14 #
15 #
16 # provide project path where sites are kept
17 project_path = 'D:/NGRI/FIELD RAW DATA/demo/' #<----- Line selected
18
19 os.chdir(project_path)
20 sites = [d for d in os.listdir('.') if os.path.isdir(d)]
21 # all site names are stored in variable sites
22 print('Sites in the project are: ')
23 print(sites)
24 # enter site = input("Enter the site: ")
25 selectedsite = 'KL33A'
26 siteindex = sites.index(selectedsite)
27 measid = mproc.measid(siteindex)
28 del siteindex
29 os.chdir(project_path+selectedsite)
30 all_meas = [d for d in os.listdir('.') if os.path.isdir(d)]
31
32 for i in range(len(all_meas)):
33     ...

```

Source | Console | Object

Usage

Here you can get help of any object by pressing **Ctrl+H** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object name to activate this behavior in **Preferences > Help**.

New to Spyder? Read our [tutorial](#)

Console 1/A X

Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 7.29.0 -- An enhanced Interactive Python.
In [1]:

LSP Python: ready cond: base (Python 3.9.7) Line 147, Col 1 UTF-8 CRLF RW Mem 52%

Figure 10: Give path and run the program

Once we run the program, the names of all sites will be displayed in the console (Figure 11). Then enter the site name to be processed and click ‘Enter’.

```

D:\Pyth\demo - Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\Pyth\demo
demo
  -> callfiles
    coh.py
    coherence.py
    mahaDist.py
    main_script.py
    main_script_I
    MPD.py
    mproc.py
    mprocRR.py
    mprocRRR.py
    plotData.py
    plotMD.py
    plotMPD.py
    tipper.py
    tipperR.py
    write2dI_da
    write2dI_he
    write2dI_co
    write2dI_py

```

Run

```

main_script.py X Run
1  # -*- coding: utf-8 -*-
2  """
3  Created on Mon May  4 16:49:35 2020
4
5  @author: AJITHABH
6  """
7  import mproc, coh, coherence, tipper, mahaDist
8  from scipy import signal
9  from matplotlib import pyplot as plt
10 import numpy as np
11 import os
12 import time
13 import math
14 #
15 #
16 # provide project path where sites are kept
17 project_path = 'D:/NGRI/FIELD RAW DATA/demo/' #<----- Line selected
18
19 os.chdir(project_path)
20 sites = [d for d in os.listdir('.') if os.path.isdir(d)]
21 # all site names are stored in variable sites
22 print('Sites in the project are: ')
23 print(sites)
24 # enter site = input("Enter the site: ")
25 selectedsite = 'KL33A'
26 siteindex = sites.index(selectedsite)
27 measid = mproc.measid(siteindex)
28 del siteindex
29 os.chdir(project_path+selectedsite)
30 all_meas = [d for d in os.listdir('.') if os.path.isdir(d)]
31
32 for i in range(len(all_meas)):
33     ...

```

Source | Console | Object

Usage

Here you can get help of any object by pressing **Ctrl+H** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object name to activate this behavior in **Preferences > Help**.

New to Spyder? Read our [tutorial](#)

Console 1/A X

Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.
IPython 7.29.0 -- An enhanced Interactive Python.
In [1]: runfile('D:/Pyth/demo/main_script.py', wdir='D:/Pyth/demo')
Sites in the project are:
['KL33A', 'LU16', 'LU33A']
Enter the site: KL33A

LSP Python: ready cond: base (Python 3.9.7) Line 36, Col 31 UTF-8 CRLF RW Mem 50%

Figure 11: Enter the site name from the list

Then all measurements in the selected site will be displayed. Give the measurement number as input. For example, I selected number 16 for the processing in Figure 12.

```

D:\Pyth\demo - Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\Pyth\demo\main_script.py
main_script.py
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon May 4 16:49:35 2020
4
5 @author: AJITHABH
6
7 import mproc, coh, coherency, tipper, mahaDist
8 from scipy import signal
9 from matplotlib import pyplot as plt
10 import numpy as np
11 import os
12 import time
13 import math
14 #
15 # provide project path where sites are kept
16 #
17 # project_path = 'D:/NGRI/FIELD RAW DATA/demo/'
18 project_path = 'D:/NGRI/FIELD RAW DATA/demo/'
19 #
20 #
21 os.chdir(project_path)
22 sites = [d for d in os.listdir('.') if os.path.isdir(d)]
23 print('Sites in the project are: ')
24 print(sites)
25 selectedsite = input('Enter the site: ')
26 siteindex = sites.index(selectedsite)
27 measid = mproc.measid(siteindex)
28 del siteindex
29 os.chdir(project_path+selectedsite)
30 all_meas = [d for d in os.listdir('.') if os.path.isdir(d)]
31 for i in range(len(all_meas)):
32

```

Sites in the project are:
 ['KL33A', 'LU16', 'LU33A']
 Enter the site: KL33A
 [[0, 'meas_2019-08-29_09-01-00'],
 [1, 'meas_2019-08-29_09-04-00'],
 [2, 'meas_2019-08-29_09-11-00'],
 [3, 'meas_2019-08-29_09-18-00'],
 [4, 'meas_2019-08-29_09-25-00'],
 [5, 'meas_2019-08-29_14-06-00'],
 [6, 'meas_2019-08-29_14-13-00'],
 [7, 'meas_2019-08-29_14-36-00'],
 [8, 'meas_2019-08-29_14-39-00'],
 [9, 'meas_2019-08-29_15-01-00'],
 [10, 'meas_2019-08-29_15-23-00'],
 [11, 'meas_2019-08-29_16-05-00'],
 [12, 'meas_2019-08-29_16-12-00'],
 [13, 'meas_2019-08-29_16-49-00'],
 [14, 'meas_2019-08-29_16-52-00'],
 [15, 'meas_2019-08-29_21-54-00'],
 [16, 'meas_2019-08-29_21-57-00'],
 [17, 'meas_2019-08-29_22-19-00']]
 Select measurement: 16

Figure 12: Enter the measurement number

Then some details about the measurement such as sampling frequency, coil numbers, length of time series, window length, and number of stacks will be displayed Figure 13. Then the processing will be started.

```

D:\Pyth\demo - Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\Pyth\demo\main_script.py
main_script.py
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon May 4 16:49:35 2020
4
5 @author: AJITHABH
6
7 import mproc, coh, coherency, tipper, mahaDist
8 from scipy import signal
9 from matplotlib import pyplot as plt
10 import numpy as np
11 import os
12 import time
13 import math
14 #
15 # provide project path where sites are kept
16 #
17 # project_path = 'D:/NGRI/FIELD RAW DATA/demo/'
18 project_path = 'D:/NGRI/FIELD RAW DATA/demo/'
19 #
20 #
21 os.chdir(project_path)
22 sites = [d for d in os.listdir('.') if os.path.isdir(d)]
23 print('Sites in the project are: ')
24 print(sites)
25 selectedsite = input('Enter the site: ')
26 siteindex = sites.index(selectedsite)
27 measid = mproc.measid(siteindex)
28 del siteindex
29 os.chdir(project_path+selectedsite)
30 all_meas = [d for d in os.listdir('.') if os.path.isdir(d)]
31 for i in range(len(all_meas)):
32

```

Basic details

Progress →

NT site: KL33A
 Measurement directory: meas_2019-08-29_21-57-00
 fs=4096.0 Hz
 Sensor numbers:
 {'Hx': [300], 'Hy': [301], 'Hz': [302]}
 Length of time series = 4915200

 Unused variables deleted.

 Window length selected: 16384
 Time series overlap: 50%
 No. of stacks: 599

 Band averaging over target frequencies:
 35|| , 17/599 [08:09:05;13, 1.86it/s]

Figure 13: Processing..

After completing the processing, figure showing apparent resistivity and phase curves, coherency values and tipper data will be plotted and can be seen in ‘Plots’ section (Figure 14).

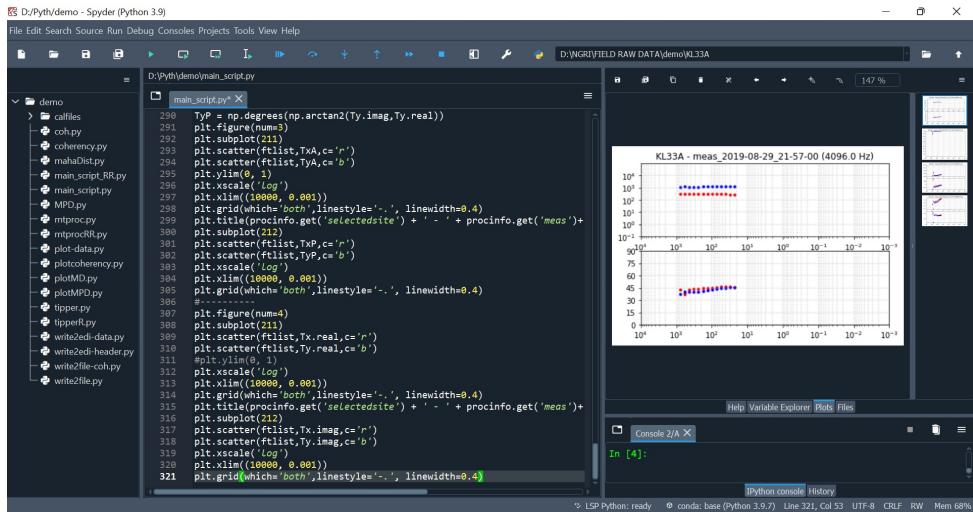


Figure 14: Output figures

4.2 Creating EDI file

The data is processed band wise. After completing processing for all measurements, the data should be joined to create the EDI files. One site may have many measurements. So, we have to save data in text files.

Create a folder anywhere. For example, a folder named ‘Outputs’ (as in Figure 15). Inside that folder, make a folder with site name. Create a folder named ‘bands’ inside site folder (Figure 15).

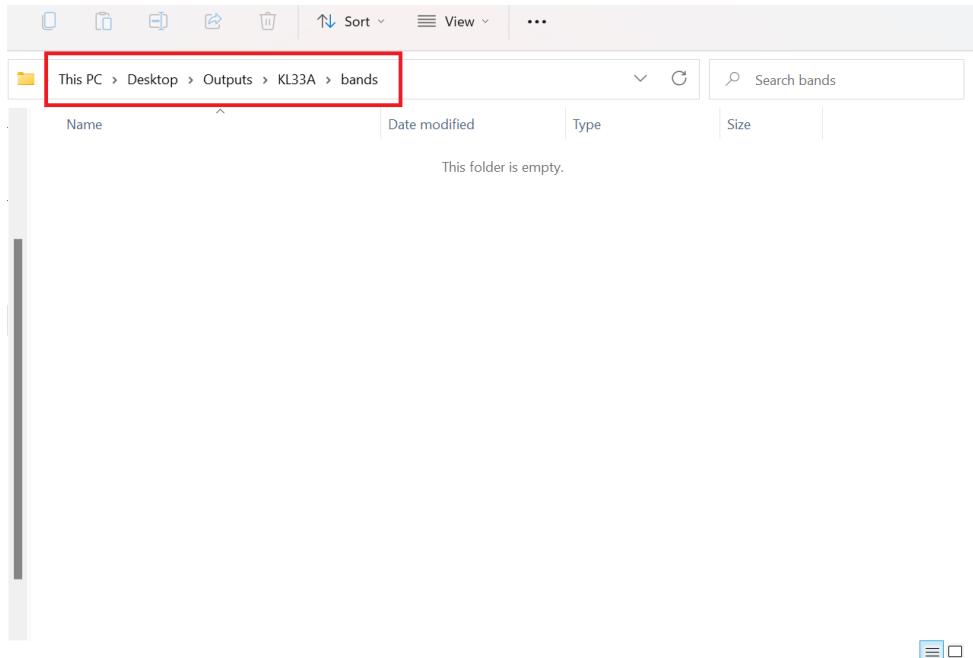


Figure 15: Create a folder to save processed data

Now come to Spyder and open ‘write2file.py’ script. Give the path to bands folder in

the variable ‘f’. Give sampling frequency as file name. For example ‘4096Hz.txt’. Select all script lines using ‘Ctrl+A’ and right click. Select ‘Run section or current line’ (Figure 16).

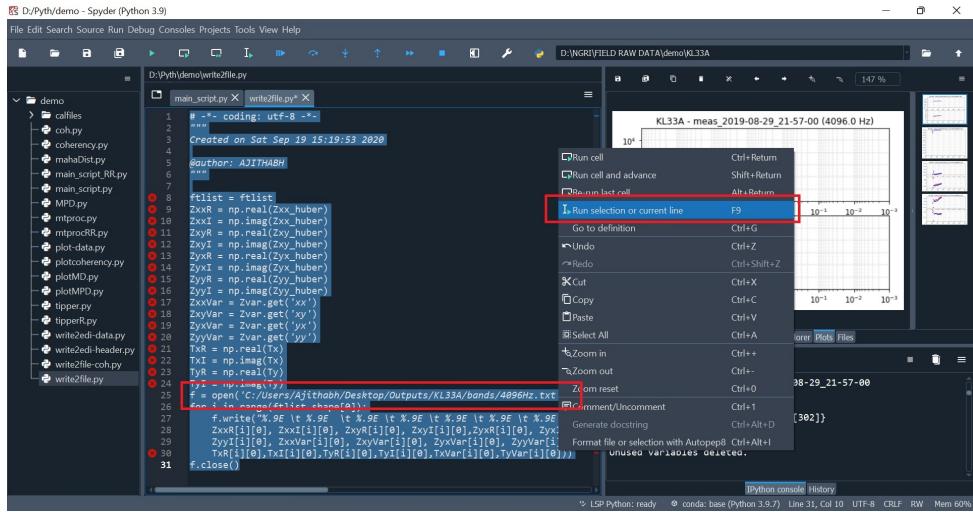


Figure 16: Save data in file

Now save header information to a file in site output folder. It is required for EDI file creation. Open ‘write2edi-header.py’ file and give path to sites folder (output) in variable ‘f’. Select all lines, right click and select ‘Run section or current line’ (Figure 17)

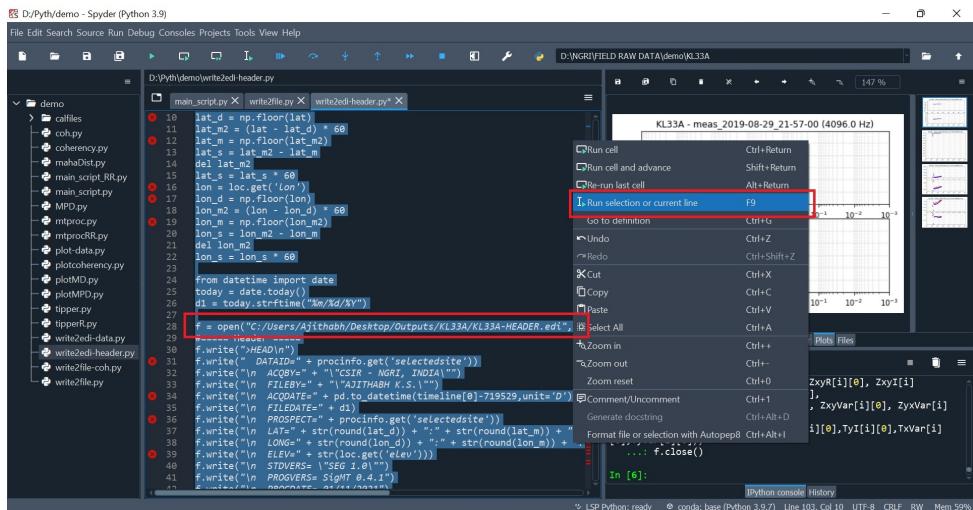


Figure 17: Save header information

After executing processing for a measurement, always close ‘Console’ (Figure 18).

The screenshot shows the Spyder Python IDE interface. On the left, there's a file tree for a 'demo' folder containing various Python scripts like 'coh.py', 'coherence.py', 'mahaDist.py', etc. In the center-left, two code editors are open: 'main_script.py' and 'write2file.py'. The code in 'main_script.py' includes imports for numpy, pandas, and procinfo, along with logic for calculating latitudes and longitudes from a location object. It also includes code for writing to a file named 'all.txt'. The code in 'write2file.py' is a template for writing data to a file. On the right side, there's a plot window titled 'KL33A - meas_2019-08-29_21-57-00 (4096.0 Hz)' showing a log-log plot of data. Below the plot is a 'Console 2/A' window with some command-line output.

Figure 18: Close console after processing a measurement

Similarly, I processed for 1024 Hz measurement and save data in bands folder of KL33A. Now the measurement data can be seen in the folder (Figure 19). Create a text file named 'all.txt'.

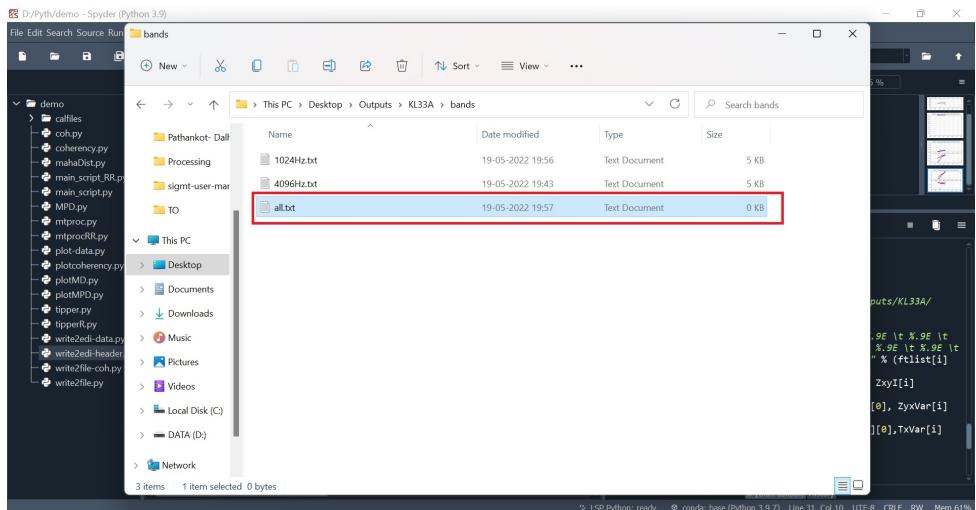


Figure 19: Create a file all.txt

Copy the data from 4096Hz.txt file and 1024Hz.txt file to all.txt as in Figure 20. Always keep higher frequency data in the top.

The screenshot shows a Notepad window with the title 'all.txt - Notepad'. The content of the file is a large block of numerical data, mostly in scientific notation. Several lines in the data are highlighted in red, specifically: '4096', '1024', and '1'. The Notepad interface includes standard menu options like File, Edit, View, and a status bar at the bottom showing 'Ln 5, Col 1', 'Windows (CR/LF)', 'UTF-8', and a date/time stamp '19-05-2022'.

Figure 20: Copy data to all.txt file

Now, again go to Spyder and open ‘write2edi-data.py’ file. Give file path to ‘all.txt’ and path of sites folder in variables ‘edifilename’ and ‘f’ (Figure 21).

The screenshot shows the Spyder IDE interface with the file 'write2edi-data.py' open. The code defines a variable 'edifilename' pointing to a CSV file and reads it into a pandas DataFrame 'data'. It then iterates through the data to extract specific columns and rows into arrays 'zxyr', 'zyxr', 'zxyvar', 'zyxvar', 'zyyvar', and 'zyyvar'. A context menu is open over the code, with the option 'I. Insert selection or current line' highlighted. The Spyder interface includes a sidebar with project files, a console at the bottom, and a status bar at the bottom right.

Figure 21: Write data as in EDI

Now, two files are existing in sites folder (Figure 22). Copy data in the file KL33A- DATA.edi to the end of header information in KL33A-HEADER.edi file (Figure 23). Please correct ‘NFREQ’ variable with actual number of frequencies in the data (Figure 23). Now, KL33A-HEADER.edi is a standard EDI file ready to use.

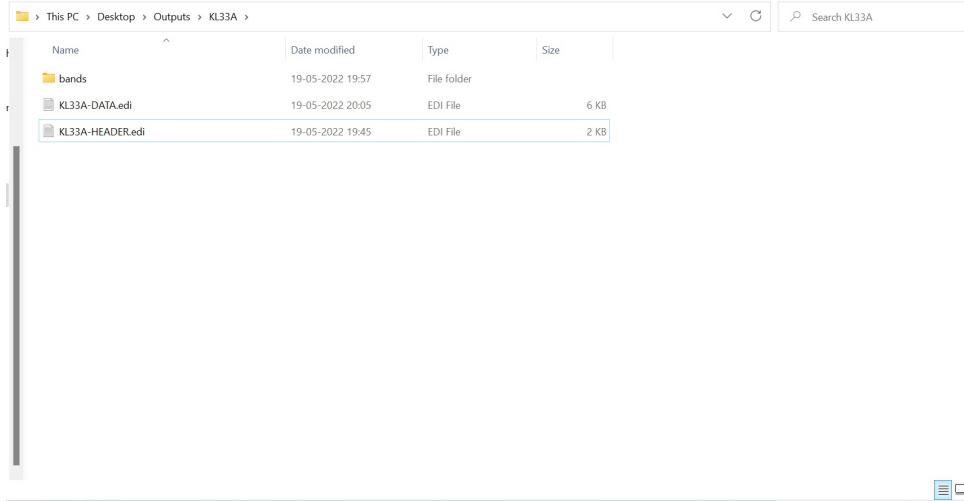


Figure 22: Files ready to create EDI

```
*KL33A-HEADER.edi - Notepad
File Edit View
>HMEAS ID=4.001 CHTYPE=HY X=0.00000000E+00
Y=0.00000000E+00 Z=0.00000000E+00
ACQCHAN="ADU07/MFS06 /0/" GAIN=1 MEASDATE=08/29/2019
AZH=9.00000000E+01 DIP=0.00000000E+00
SENSOR=MFS06 /
>HMEAS ID=5.001 CHTYPE=HZ X=0.00000000E+00
Y=0.00000000E+00 Z=0.00000000E+00
ACQCHAN="ADU07/MFS06 /0/" GAIN=1 MEASDATE=08/29/2019
AZH=0.00000000E+00 DIP=0.00000000E+00
SENSOR=MFS06 /
>=HTSCT
>SECTID=KL33A
NFREQ=13
HX=3.001
HY=4.001
HZ=5.001
EX=1.001
EY=2.001
>FREQ //17
5.7103854095E+02 4.258777131E+02 3.176174875E+02 2.368775478E+02 1.766621010E+02
5.317537193E+02 9.826127078E+01 3.28275869E+01 5.465392565E+01 4.076062685E+01
3.039907346E+01 2.267147831E+01 1.698827615E+01 1.261810854E+01 9.404556446E+00
7.013871582E+00 5.230910660E+00
>ZXXR //17
-7.023137803E+01 1.020347167E+01 1.315706979E+01 1.655562791E+01 1.870547240E+01
2.023157685E+01 1.7840000015E+01 2.158549529E+01 2.051765238E+01 1.725275253E+01
1.361852426E+01 5.388805539E+00 -4.158872013E-01 -2.430117559E+00 -2.661894916E+00
-1.072436996E-02 5.565941568E+00
Ln 58, Col 11
100% Windows (CRLF) UTF-8
```

Figure 23: Edit NFREQ and save EDI

4.3 Data selection tools

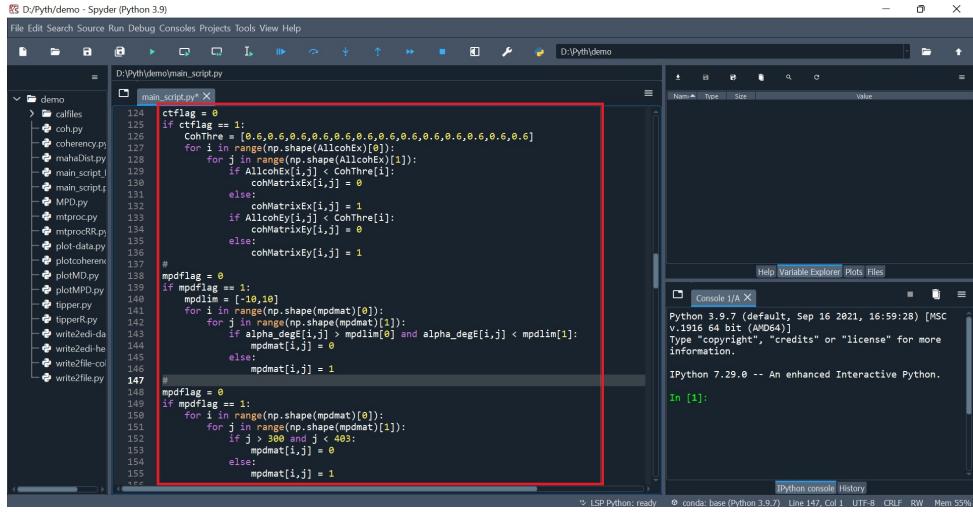


Figure 24: Data selection tools section

In Figure 24, we can see the data selection tools part of package. If `ctflag` is 1, the coherency threshold will be activated. The coherency threshold value can be provide in ‘`CohTre`’ variable for each target frequency. You can see the coherency values for each target frequency using the ‘`plotcoherency.py`’ script. Give `Z_all = bandavg.get('Zxy_single')` to see coherency of xy component and `Z_all = bandavg.get('Zyx_single')` for yx component. Once you run the script, coherency plots will be displayed for each target frequency as in Figure 26. By analysing the values in the figure, you can set cohrency threshold values.

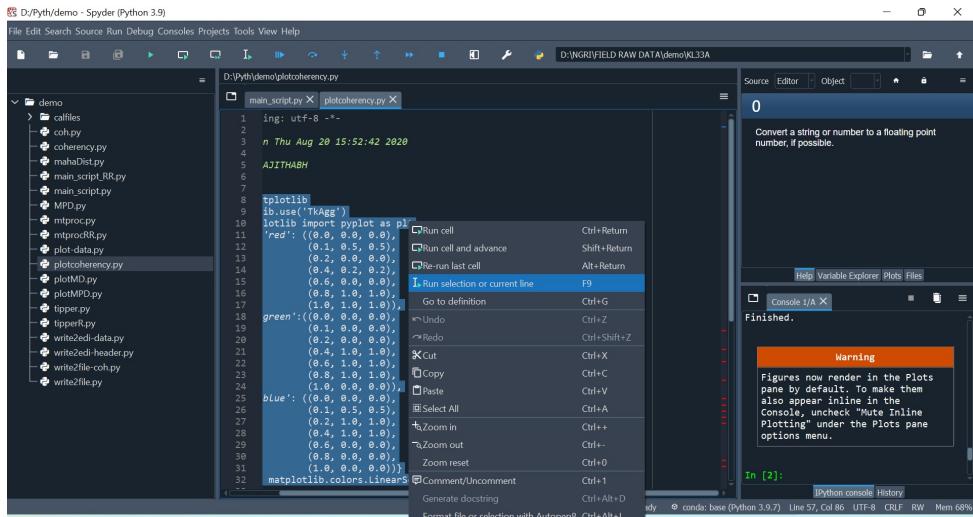


Figure 25: Run codes

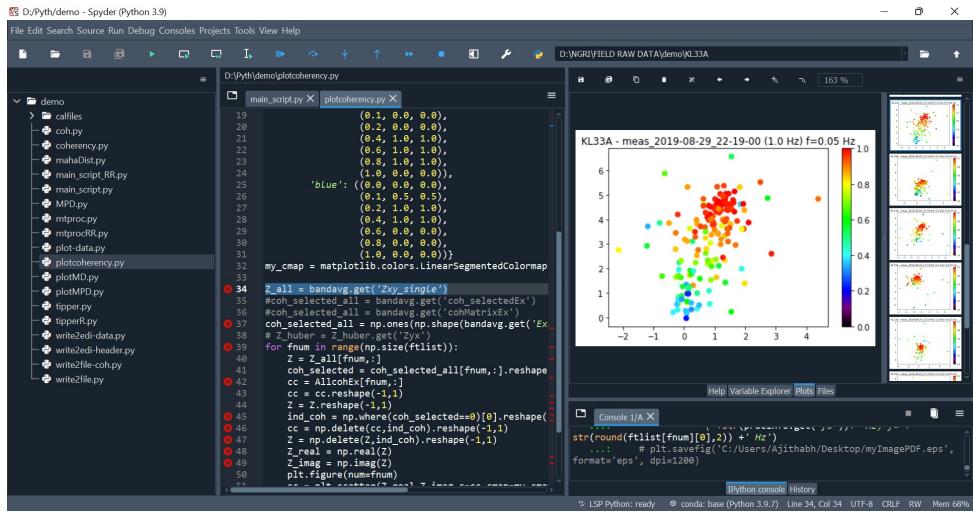


Figure 26: Coherency values

As similar in the case of coherency threshold, run scripts in ‘plot-pd.py’ to plot polarization directions for each stacks for all target frequencies (Figure 27). The top panel in the plot shows magnetic polarization directions and bottom panel shows electric polarization directions. Analysing the plots, we can identify the polarized segments. Polarization directions can be selected in two ways. We can select a range of polarization direction to be discarded in the case 1. All stacks with polarization direction values [-10,10] will be discarded in this case. In case you need to use magnetic polarization direction, use ‘alpha_degH’ and for electric polarization direction, use ‘alpha_degE’ (as in Figure 28).

Next is to select stacks, give a range of stacks to be discarded in this case. In the example (Figure 28), the stacks between 300 and 403 will be discarded from processing.

‘mpdfflag’ should be 1 to perform the tasks.

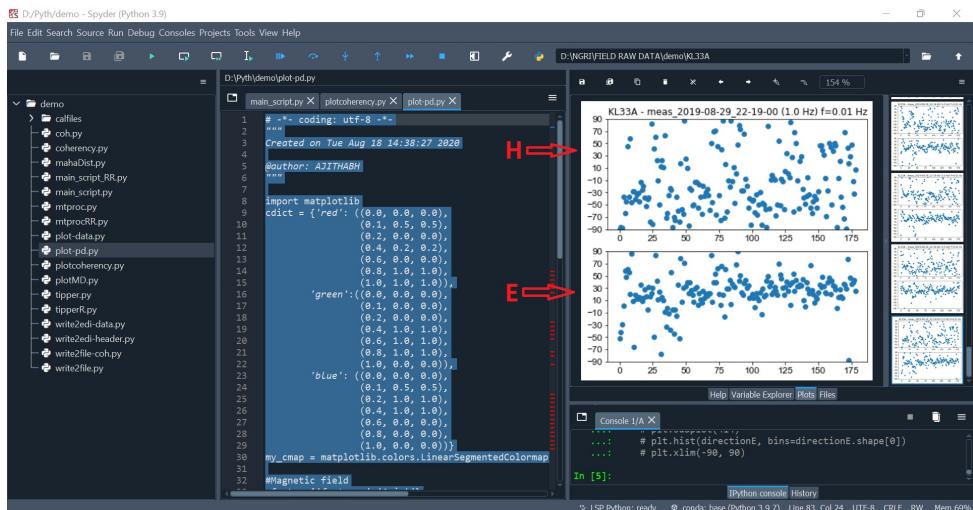


Figure 27: Polarization directions

```

D:\Pyth\demo - Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\Pyth\demo
demo
|- callfiles
|- coh.py
|- coherence.py
|- mahaDist.py
|- main_script_RR.py
|- main_script.py
|- MDP.py
|- mtproc.py
|- mtprocRR.py
|- plot-data.py
|- plotMD.py
|- plotMDD.py
|- tipper.py
|- tipperR.py
|- write2edi-data.py
|- write2edi-header.py
|- write2file-coh.py
|- write2file.py

D:\Pyth\demo\main_script.py
124 cohThre = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
125 for i in range(np.shape(AlcohEx)[0]):
126     for j in range(np.shape(AlcohEx)[1]):
127         if AlcohEx[i,j] < cohThre[i]:
128             cohMatrixEx[i,j] = 0
129         else:
130             cohMatrixEx[i,j] = 1
131         if AlcohEx[i,j] < cohThre[i]:
132             cohMatrixEy[i,j] = 0
133         else:
134             cohMatrixEy[i,j] = 1
135 #
136 mpdFlag = 0
137 if mpdFlag == 1:
138     mpdlim = [10,10]
139     for i in range(np.shape(mpdmat)[0]):
140         for j in range(np.shape(mpdmat)[1]):
141             if alpha_degE[i,j] > mpdlim[0] and alpha_degE[i,j] < mpdlim[1]:
142                 mpdmat[i,j] = 0
143             else:
144                 mpdmat[i,j] = 1
145 #
146 mpdFlag = 0
147 if mpdFlag == 1:
148     for i in range(np.shape(mpdmat)[0]):
149         for j in range(np.shape(mpdmat)[1]):
150             if j > 360 and j < 408:
151                 mpdmat[i,j] = 0
152             else:
153                 mpdmat[i,j] = 1
154
155

```

Figure 28: Magnetic or Electric Polarization direction

Once data selection constraints are set, just run a part of code as highlighted (to end of the script) in figure 29.

```

D:\Pyth\demo - Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
D:\Pyth\demo
demo
|- callfiles
|- coh.py
|- coherence.py
|- mahaDist.py
|- main_script_RR.py
|- main_script.py
|- MDP.py
|- mtproc.py
|- mtprocRR.py
|- plot-data.py
|- plotMD.py
|- plotMDD.py
|- tipper.py
|- tipperR.py
|- write2edi-data.py
|- write2edi-header.py
|- write2file-coh.py
|- write2file.py

D:\Pyth\demo\main_script.py
104 ##### Timer calculation #####
105 [time_start, timer_end] = tipper.tipspall(bandavg)
106 timer_end = time.time()
107 print('In[closed time: ' + str(timer_end - timer_start)+ 's')
108 del timer_start, timer_end
109 print('Finished.')
110 mahaWtx, Tx_mcd_mean, Txmaha_robust = tipper.mcd(TxAll)
111 mahaWty, Ty_mcd_mean, Tymaha_robust = tipper.mcd(TyAll)
112 ##### Timer calculation #####
113 # Coherence Threshold
114 cohMatrixEx = np.ones(np.shape(bandavg.get('ExExc')),dtype=float)
115 cohMatrixY = np.ones(np.shape(bandavg.get('ExExc')),dtype=float)
116 mpdmat = np.ones(np.shape(bandavg.get('ExExc')),dtype=float)
117 AlcohEx = coherence.coheBy(bandavg)
118 AlcohY = coherence.coheBy(bandavg)
119 alpha_degE, alpha_degE = mtproc.mpvalues(bandavg)
120 #
121 #
122 ctflag = 0
123 if ctflag == 1:
124     cohThre = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
125     for i in range(np.shape(AlcohEx)[0]):
126         for j in range(np.shape(AlcohEx)[1]):
127             if AlcohEx[i,j] < cohThre[i]:
128                 cohMatrixEx[i,j] = 0
129             else:
130                 cohMatrixEx[i,j] = 1
131             if AlcohEx[i,j] < cohThre[i]:
132                 cohMatrixEy[i,j] = 0
133             else:
134                 cohMatrixEy[i,j] = 1
135

```

Figure 29: Run this part of script

4.4 Remote reference

The remote reference can be done similar as above example. But run ‘main_script_RR.py’ file. First you have to enter the site you need to process and enter the measurement number. Then enter the remote site name and measurement number.