```python
from pprint import pprint
from PIL import Image
import numpy as np
import pandas as pd
import os
import matplotlib.pyplot as plt
import glob
import xml.etree.ElementTree as ET
from scipy.signal import savgol_filter
import gc
from scipy.interpolate import CubicSpline
from scipy.interpolate import interp1d

# Distance betwen moon and sun in AUs
D_AU = 1

# Solar irradiance
F0 = np.array([
  136.1259307,
  129.8781929,
  125.1457188,
  120.4566749,
  115.2187742,
  110.7989129,
  105.971862,
  102.2853476,
  98.83159112,
  95.00990644,
  91.72241746,
  88.63043389,
  85.44216416,
  83.09659958,
  80.7461688,
  77.99745659,
  75.43755054,
  72.53298554,
  70.30310472,
  67.71506702,
  65.53063581,
  63.51647332,
  61.49193881,
  59.39769145,
  57.24811211,
  55.56974549,
  53.96628612,
  52.39858882,
  50.94286582,
  49.55873832,
  47.99340839,
  46.35543865,
```

```
45.11640663,
43.75374359,
42.46741487,
41.1950428,
39.93375405,
38.7480202,
37.63257797,
36.52968828,
35.48372942,
34.51571377,
33.5041102,
32.62925225,
31.80035805,
30.98128654,
30.16775831,
29.32709974,
28.56074168,
27.8298174,
27.0453247,
26.30808675,
25.51810387,
24.75010497,
24.00573968,
23.24760491,
22.51761852,
21.78398871,
21.06792047,
20.39822233,
19.7458807,
19.11661541,
18.44061437,
17.83250529,
17.26068394,
16.65126453,
16.11545704,
15.61912435,
15.1210474,
14.62910738,
14.16359209,
13.72237684,
13.31430194,
12.94713935,
12.56233275,
12.18239943,
11.79722098,
11.38810049,
11.04636914,
10.71621297,
10.38904988,
```

```
    10.06620698,
    9.753295821,
    9.46418631,
    9.201075776,
    8.960974818,
    8.732115834,
    8.508712424,
    8.28861478,
    8.070068082,
    7.850866176,
    7.629585176,
    7.417896212,
    7.21399149,
    7.014245694,
    6.819995994,
    6.637200746,
    6.463212542,
    6.291676014,
    6.122400975,
    5.952327234,
    5.785907458,
    5.631916792,
    5.48221029,
    5.338864421,
    5.183886388,
    5.053359936,
    4.941756508,
    4.835098184,
    4.719922707,
    4.619729215,
    4.511137419,
    4.407240202,
    4.306184976,
    4.210413629,
    4.117013411,
    4.012368768,
    3.918726643,
    3.824014432,
    3.725826304,
    3.646586732,
    3.564719937,
    3.488199195,
    3.397463341,
    3.32250234,
    3.262984894,
    3.190955311,
    3.122692223,
    3.056477464,
    2.991274348,
```

```
2.926566072,
2.864612339,
2.802940836,
2.743157021,
2.685370618,
2.628641884,
2.571929704,
2.517226294,
2.465127643,
2.414375576,
2.365285234,
2.316701141,
2.26923212,
2.222564505,
2.178496705,
2.135290025,
2.092826765,
2.051565701,
2.010893773,
1.971470582,
1.932492639,
1.893925453,
1.853239032,
1.814419696,
1.780829606,
1.751599126,
1.715922793,
1.680125966,
1.647791753,
1.621454182,
1.593640531,
1.560460708,
1.532378246,
1.507178355,
1.480349348,
1.454525518,
1.426003985,
1.40026592,
1.376814112,
1.351395724,
1.327241488,
1.303320437,
1.279240078,
1.255715058,
1.232621586,
1.209534773,
1.186777237,
1.163774025,
1.141839466,
```

```
1.121354795,
1.102697582,
1.084984542,
1.06779729,
1.050654559,
1.034116451,
1.018239678,
1.003106371,
0.987228033,
0.971082552,
0.954532246,
0.938549781,
0.922761605,
0.90746215,
0.892772367,
0.876952832,
0.86169586,
0.846904043,
0.832961745,
0.820193322,
0.808495532,
0.796418017,
0.784036511,
0.771772032,
0.760169612,
0.74902997,
0.737997332,
0.727055348,
0.716477866,
0.704633464,
0.691770452,
0.681177697,
0.668685204,
0.6563386,
0.643784606,
0.630929839,
0.618670348,
0.605670184,
0.593191697,
0.582320158,
0.571630629,
0.561438106,
0.551831735,
0.542986524,
0.534529199,
0.526707332,
0.518722109,
0.511109087,
0.50373316,
```

```
  0.496221855,
  0.489530981,
  0.482582186,
  0.475974536,
  0.469794569,
  0.463575699,
  0.458286546,
  0.452850271,
  0.447197638,
  0.441572082,
  0.43580287,
  0.430755766,
  0.425717099,
  0.420589447,
  0.41588213,
  0.410468477,
  0.405233536,
  0.399887123,
  0.394668014,
  0.389642973,
  0.384580319,
  0.379611238,
  0.374544041,
  0.369613524,
  0.364863435,
  0.360132602,
  0.355533758,
  0.350967069,
]).reshape(256,1,1)

class utils:
    @staticmethod
    def _extract_sequence_numbers(file_path):
        """
        Extracts the text content of <sequence_number> elements from
an XML file.

        Args:
            file_path (str): The path to the XML file.

        Returns:
            list: A list of text contents from <sequence_number>
elements.
        """
        # Parse the XML file
        tree = ET.parse(file_path)
        root = tree.getroot()

        # Find all sequence_number elements
        sequence_numbers =
```

```
root.findall('.//{http://pds.nasa.gov/pds4/pds/v1}elements')

        # Extract the text content from each <sequence_number> element
        return [int(sequence_number.text) for sequence_number in
sequence_numbers]
    @staticmethod
    def _find_xml_files(base_path):
        """
        Finds all XML files matching the pattern
/data/calibrated/*/*.xml within the given base path.

        Args:
            base_path (str): The base directory path where the search
begins.

        Returns:
            list: A list of paths to the matching XML files.
        """
        # Define the search pattern
        pattern = os.path.join(base_path, 'data', 'calibrated', '*',
'*.xml')

        # Use glob to find all files matching the pattern
        matching_files = glob.glob(pattern)
        matching_files.sort()
        return matching_files

    @staticmethod
    def _find_qub_files(base_path):
        """
        Finds all .qub files matching the pattern
/data/calibrated/*/*.qub within the given base path.

        Args:
            base_path (str): The base directory path where the search
begins.

        Returns:
            list: A list of paths to the matching .qub files.
        """
        # Define the search pattern
        pattern = os.path.join(base_path, 'data', 'calibrated', '*',
'*.qub')

        # Use glob to find all files matching the pattern
        matching_files = glob.glob(pattern)
        matching_files.sort()
        return matching_files
    @staticmethod
    def _get_image_array(qub_path,shape):
```

```
        """
        Args:
            qub_path : path to the .qub file
            shape : shape of the image of form (channels,height,width)
        returns:
            a numpy array reshaped in the shape provided
        """
        with open(qub_path, 'rb') as f:
            img = np.frombuffer(f.read(),
dtype=np.float32).reshape(*shape)
        return img

    @staticmethod
    def get_image(base_path):
        """
        Args:
            base_path : Base directory path to the data. Which means
the path to directory which contains the browse, data geometyr and
miscellaneous folder
        returns:
            A list of images with the radiance resized and resahped.
To visualize it you need to do min max scaling and stuff
        """
        xml_files = utils._find_xml_files(base_path)
        image_files = utils._find_qub_files(base_path)
        shapes = [utils._extract_sequence_numbers(xml_file) for
xml_file in xml_files]
        images = [utils._get_image_array(qub_path,shape) for
qub_path,shape in zip(image_files,shapes)]
        return images
    @staticmethod
    def _read_misc_files_into_df(file_path):
        with open(file_path) as f:
            data = f.read()
        rows = data.split('\n')
        parsed_list = [' '.join(row.split()).split() for row in rows]
        return pd.DataFrame(parsed_list)

    @staticmethod
    def _get_misc_files(base_path):
        """
        Args:
            base_path : Base directory path to the data. Which means
the path to directory which contains the browse, data geometyr and
miscellaneous folder
        returns:
            A dictionary of miscellaneous files converted to the
dataframe
        """
```

```
        pattern = os.path.join(os.path.join(base_path,
'miscellaneous', 'calibrated', '*', '*.*'))
        matching_files = glob.glob(pattern)
        matching_files.sort()

        dfs = {}
        for file_path in matching_files:
            print(file_path)
            _, file_extension = os.path.splitext(file_path)
            dfs[file_extension] =
utils._read_misc_files_into_df(file_path).dropna()
        return dfs
    @staticmethod
    def get_misc_files(base_path):
        """
        Args:
            base_path : Base directory path to the data. Which means
the path to directory which contains the browse, data geometyr and
miscellaneous folder
        returns:
            A dictionary of miscellaneous files converted to the
dataframe
        """

        dfs = utils._get_misc_files(base_path)
        dfs['.spm'] = utils._process_spm_df(dfs['.spm'])
        dfs['.oat'] = utils._process_oat_df(dfs['.oat'])
        return dfs
    @staticmethod
    def convert_to_reflectance(data, solar_zenith_angle):
        return (np.pi * data)/(np.cos(solar_zenith_angle * np.pi /
180) * F0 * D_AU**2)

    @staticmethod
    def _process_spm_df(spm_df):
        column_names = [
            "Record type", "Physical record number", "Block length",
        #     "Year",
            "Month",
            "Date",
            "Hour",
            "Minute",
            "Second",
            "Millisec", "Satellite position X", "Satellite position
Y",
            "Satellite position Z", "Satellite velocity X-dot",
            "Satellite velocity Y-dot", "Satellite velocity Z-dot",
            "Phase angle", "Sun aspect", "Sun Azimuth",
            "Sun Elevation",
```

```
        ]

        # Assign the new column names
        spm_df.columns = column_names

        # Convert the columns to the appropriate data types
        spm_df["Record type"] = spm_df["Record type"].astype(str)
        spm_df["Physical record number"] = spm_df["Physical record
number"].astype(np.int32)
        spm_df["Block length"] = spm_df["Block
length"].astype(np.int32)
        # spm_df['Year'] = spm_df['Year'].astype(np.int32)
        spm_df['Month'] = spm_df['Month'].astype(np.int32)
        spm_df['Date'] = spm_df['Date'].astype(np.int32)
        spm_df['Hour'] = spm_df['Hour'].astype(np.int32)
        spm_df['Minute'] = spm_df['Minute'].astype(np.int32)
        spm_df['Second'] = spm_df['Second'].astype(np.int32)
        spm_df['Millisec'] = spm_df['Millisec'].astype(np.int32)
        # For "Time in UTC", assuming it is in the format
'YYYYMMDDHHMMSS', convert to datetime

        spm_df["Satellite position X"] = spm_df["Satellite position
X"].astype(np.float32)
        spm_df["Satellite position Y"] = spm_df["Satellite position
Y"].astype(np.float32)
        spm_df["Satellite position Z"] = spm_df["Satellite position
Z"].astype(np.float32)
        spm_df["Satellite velocity X-dot"] = spm_df["Satellite
velocity X-dot"].astype(np.float32)
        spm_df["Satellite velocity Y-dot"] = spm_df["Satellite
velocity Y-dot"].astype(np.float32)
        spm_df["Satellite velocity Z-dot"] = spm_df["Satellite
velocity Z-dot"].astype(np.float32)
        spm_df["Phase angle"] = spm_df["Phase
angle"].astype(np.float32)
        spm_df["Sun aspect"] = spm_df["Sun aspect"].astype(np.float32)
        spm_df["Sun Azimuth"] = spm_df["Sun
Azimuth"].astype(np.float32)
        spm_df["Sun Elevation"] = spm_df["Sun
Elevation"].astype(np.float32)
        # spm_df["Orbit Limb Direction"] = spm_df["Orbit Limb
Direction"].astype(int)
        return spm_df
    @staticmethod
    def _process_oat_df(oat_df):
        columns_names = [
            "Record type",
            "Physical record number in this file",
            "Block length in bytes",
            "Month",
```

```
            "Date",
            "Hour",
            "Minute",
            "Second",
            "Millisec",
            "Lunar Position X (kms) - J2000 Earth Centre Frame",
            "Lunar Position Y (kms) - J2000 Earth Centre Frame",
            "Lunar Position Z (kms) - J2000 Earth Centre Frame",
            "Satellite position X (kms) - Note-3",
            "Satellite position Y (kms) - Note-3",
            "Satellite position Z (kms) - Note-3",
            "Satellite velocity X-dot (kms/sec) - Note-3",
            "Satellite velocity Y-dot (kms/sec) - Note-3",
            "Satellite velocity Z-dot (kms/sec)  - Note-3",
            "Altitude Inertial Q1",
            "Altitude Inertial Q2",
            "Altitude Inertial Q3",
            "Altitude Inertial Q4",
            "Earth Fixed IAU frame Q1",
            "Earth Fixed IAU frame Q2",
            "Earth Fixed IAU frame Q3",
            "Earth Fixed IAU frame Q4"
            "Lunar Fixed IAU frame Q1",
            "Lunar Fixed IAU frame Q2",
            "Lunar Fixed IAU frame Q3",
            "Lunar Fixed IAU frame Q4",
            "Latitude of sub-satellite point (deg)",
            "Longitude of sub-satellite point (deg)",
            "Solar Azimuth",
            "Solar Elevation",
            "Latitude  (deg)",
            "Longitude (deg)",
            "Satellite altitude (kms)",
            "Angle between +Roll and Velocity Vector",
            "Eclipse Status - Note-4",
            "Emission Angle",
            "Sun Angle w.r.t -ve Yaw (Phase angle)",
            "Angle between +Yaw and Nadir",
            "Slant Range (Km)",
            "Orbit No",
            "Solar Zenith Angle",
            "Angle between Payload FoV axis and velocity vector",
            "X  (yaw) angle",
            "Y  (roll) angle",
            "Z(pitch) angle",
        ]
        oat_df.columns = columns_names
        oat_df.iloc[:,1:9] = oat_df.iloc[:,1:9].astype(np.int32)
        oat_df.iloc[:,9:42] = oat_df.iloc[:,9:42].astype(np.float32)
```

```
        oat_df.iloc[:,42] = oat_df.iloc[:,42].astype(np.int32)
        oat_df.iloc[:,43:] = oat_df.iloc[:,43:].astype(np.float32)
        return oat_df
```

Denoising Function : Savitzky-Golay filter

```
def denoise(data, axis, window_size = 7, polyorder = 2):
    denoised_image = savgol_filter(data, window_length=window_size, 
polyorder=polyorder, axis=axis)
    return denoised_image

# data path should be root directory of the bundle
data_path = 
"/kaggle/input/isro-chandrayan-iirs/other/dataset-26/2/ch2_iir_nci_201
91217T0003248322_d_img_d18"
images = utils.get_image(data_path)
image = images[0]
misc_dfs = utils.get_misc_files(data_path)
oat_df = misc_dfs['.oat']
mean_zenith_angle = oat_df.loc[:,'Solar Zenith Angle'].mean()
reflectance_image = 
utils.convert_to_reflectance(image,mean_zenith_angle)
# reflectance image is the desired output
```

```
/kaggle/input/isro-chandrayan-iirs/other/dataset-26/2/
ch2_iir_nci_20191217T0003248322_d_img_d18/miscellaneous/calibrated/
20191217/ch2_iir_nci_20191217T0003248322_d_img_d18.lbr
/kaggle/input/isro-chandrayan-iirs/other/dataset-26/2/ch2_iir_nci_2019
1217T0003248322_d_img_d18/miscellaneous/calibrated/20191217/
ch2_iir_nci_20191217T0003248322_d_img_d18.oat
/kaggle/input/isro-chandrayan-iirs/other/dataset-26/2/ch2_iir_nci_2019
1217T0003248322_d_img_d18/miscellaneous/calibrated/20191217/
ch2_iir_nci_20191217T0003248322_d_img_d18.oath
/kaggle/input/isro-chandrayan-iirs/other/dataset-26/2/ch2_iir_nci_2019
1217T0003248322_d_img_d18/miscellaneous/calibrated/20191217/
ch2_iir_nci_20191217T0003248322_d_img_d18.spm
```

```
denoised_reflectance_image = denoise(reflectance_image, 0)

# del denoised_reflectance_image;  gc.collect()

# combined_array = np.concatenate((denoised_reflectance_image[7:62, 7, 
7], denoised_reflectance_image[72:115, 7, 7]))
# plt.plot(combined_array)
```

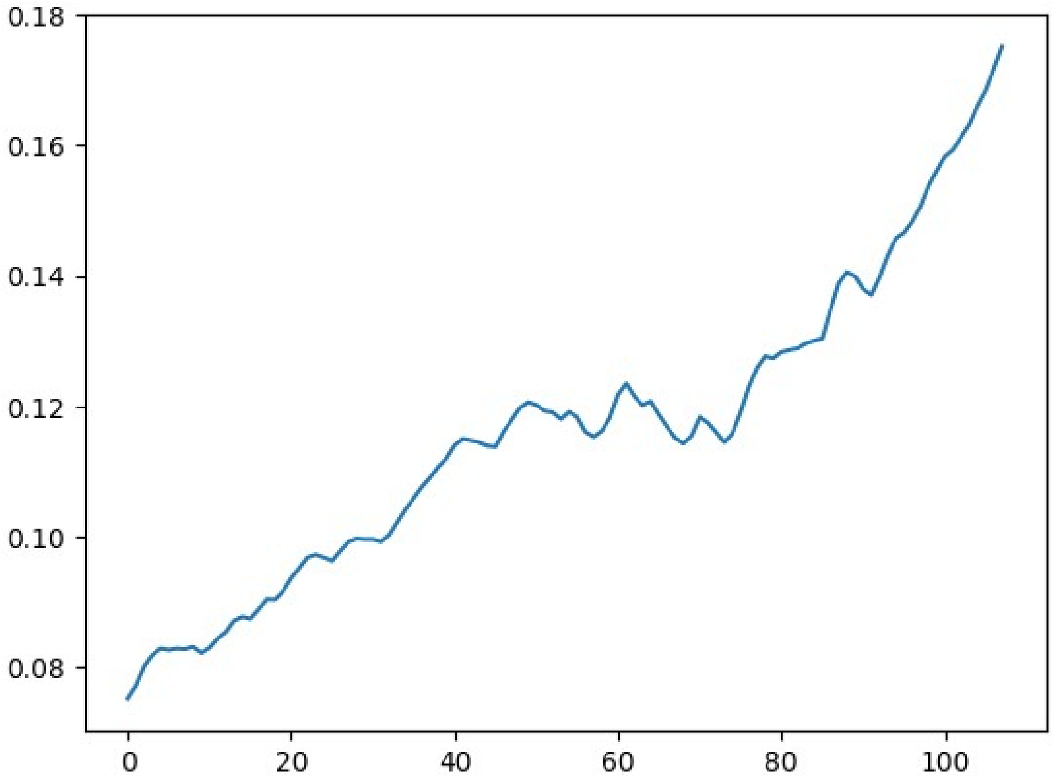
interpolation of OSF region

```
combined_array = np.concatenate((denoised_reflectance_image[7:68, 7,
7], denoised_reflectance_image[77:115, 7, 7]))

original_indices = np.concatenate((np.arange(7, 68), np.arange(77,
115)))

interpolated_indices = np.arange(7, 115)

cubic_spline = CubicSpline(original_indices, combined_array)

interpolated_array = cubic_spline(interpolated_indices)
```

add noise

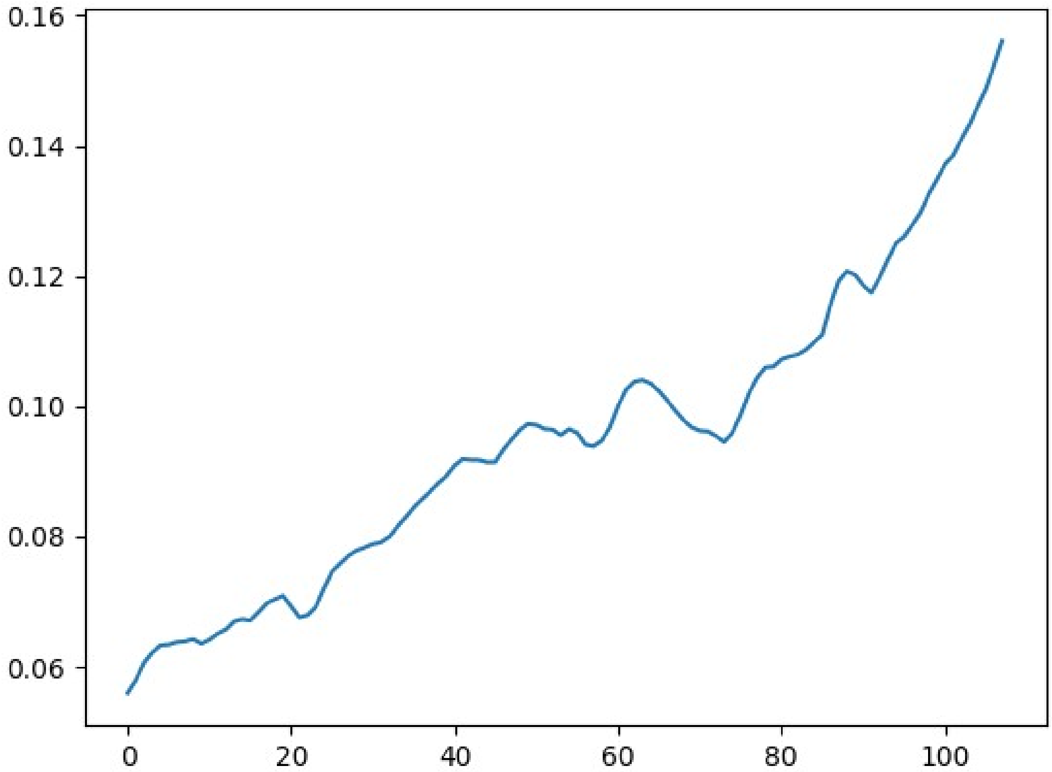
```
def add_noise(data, max_noise):

    # Add noise to reflectance data
#     noise = np.random.normal(0,
0.005,size=reflectance_vectors['bir1lm054_Breccia__Lunar_Feldspathic_B
reccia'].shape)
#     reflectance_noisy =
reflectance_vectors['bir1lm054_Breccia__Lunar_Feldspathic_Breccia'] +
noise

    wavelength = range(data.shape[0])
    low_freq_wavelength = np.linspace(0, data.shape[0], 30)  # Fewer
points
    low_freq_noise = np.random.normal(0, max_noise,
size=low_freq_wavelength.shape)

    # Interpolate the low-frequency noise to match the wavelength data
    interp_noise = interp1d(low_freq_wavelength, low_freq_noise,
kind='linear')
    noise = interp_noise(wavelength)
    # Plot the noisy reflectance data
    reflectance_noisy = data + noise
    return reflectance_noisy

plt.plot(denoised_reflectance_image[7:115, 1000, 200])
```

```
[<matplotlib.lines.Line2D at 0x7fc8de9fa260>]
```




```
plt.plot(interpolated_array)
```

```
[<matplotlib.lines.Line2D at 0x7fc82f8c3850>]
```




```
# interpolated_df =
pd.read_csv('/kaggle/input/interpolated-data/bir1lm044_Silicate_(Ino)_
_Pyroxene__from_lunar_basalt.csv')
# interpolated_reflectance = interpolated_df['Reflectance'].values

import glob
import os

folder_path = '/kaggle/input/interpolated-data'

csv_files = glob.glob(os.path.join(folder_path, '*.csv'))

reflectance_vectors = {}
reflectance_vectors_noise_25 = {}
reflectance_vectors_noise_5 = {}

for file in csv_files:
    filename = os.path.splitext(os.path.basename(file))[0]

    df = pd.read_csv(file)

    reflectance_vectors[filename] = df['Reflectance'].values[1:]
    reflectance_vectors_noise_25[filename] =
add_noise(df['Reflectance'].values[1:], 0.0025)
```

```
    reflectance_vectors_noise_5[filename] =
add_noise(df['Reflectance'].values[1:], 0.005)

def spectral_angle_mapper(pixel_vector, reference_vector):

    dot_product = np.dot(pixel_vector, reference_vector)

    norm_pixel = np.linalg.norm(pixel_vector)

    norm_reference = np.linalg.norm(reference_vector)

    cosine_angle = dot_product / (norm_pixel * norm_reference)

    angle = np.arccos(np.clip(cosine_angle, -1.0, 1.0))

    return angle

sam = {}
sam_noise_25 = {}
sam_noise_5 = {}

for key1, key2, key3 in zip(reflectance_vectors.keys(),
                            reflectance_vectors_noise_25.keys(),
                            reflectance_vectors_noise_5.keys()):
    sam[key1] =
spectral_angle_mapper(denoised_reflectance_image[7:115, 1000, 200],
reflectance_vectors[key1])
    sam_noise_25[key2] =
spectral_angle_mapper(denoised_reflectance_image[7:115, 1000, 200],
reflectance_vectors_noise_25[key2])
    sam_noise_5[key3] =
spectral_angle_mapper(denoised_reflectance_image[7:115, 1000, 200],
reflectance_vectors_noise_5[key3])

sam
```

```
{'c2mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.26234734824433403,
 'bir1lm046_unbrecciated_lunar_meteorite__coarse_texture':
0.1825719049585525,
 'c1lm36_Igneous__Lunar_Unbrecciated_Basalt_': 0.26076203401679204,
 'cclm35_Breccia__Lunar_Feldspathic_Breccia': 0.2556124697162266,
 'c10l01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24344711263092272,
 'c3mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.40429636322642726,
 'colm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2634747402331538,
```

'c12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.22633572765966495,
'c6mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.29701360841758273,
'bir1lm048_Mare_basalt': 0.16115619776886603,

'cama01_Anomalous_(ACANOM)__Antarctic_meteorite_of_presumed_lunar_orig
in': 0.20958037191047235,
'calm34_Breccia__Lunar_Feldspathic_Breccia': 0.28726072447473316,
'cflm34_Breccia__Lunar_Feldspathic_Breccia': 0.26812261859120573,
'cpmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.08428332167762563,
'c1lm50b_0.148_g': 0.2198313350685982,
'cclm34_Breccia__Lunar_Feldspathic_Breccia': 0.20623026487899174,
'cblm34_Breccia__Lunar_Feldspathic_Breccia': 0.21999016710333455,
'bir1lm053_Breccia__Lunar_Feldspathic_Breccia': 0.09406887848015909,
'celm13_(E)_White_end.': 0.28331982109888054,
'bir1lm044_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.2190160020992616,
'c1lm52b_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.18959337761408235,
'c1lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
0.14207530355056167,
'c1lm51_Lunar_meteorite__feldspathic_fragmental_breccia':
0.30297134113772617,
's12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2327375710615465,
'c1lm52c_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.11823011780326945,
'cclm13_Top_whitish.': 0.23963792481236104,
'cblm35_Breccia__Lunar_Feldspathic_Breccia': 0.25595245849647813,
'c1lm50c_0.148_g': 0.17180792866424074,
'sblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.22190035388583812,
'c2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2704091776335051,
's2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2566203906609711,
'bir1lm047_Mare_basalt': 0.18957737484674728,
'sslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2511727188361024,
'c1lm50_0.148_g': 0.2793097223070759,
'c1lm51c_Lunar_meteorite__feldspathic_fragmental_breccia':
0.15493113330760683,
'bir1lm054_Breccia__Lunar_Feldspathic_Breccia': 0.0835425883809933,
's10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25754242341212086,

```
 'c8mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.3718305494046039,
 'c9mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.18015207604378075,
 'c1lm52_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.2307419107048046,
 'cdlm34_Breccia__Lunar_Feldspathic_Breccia': 0.31995988418258586,
 'crlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24650914243138178,
 'cnmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.12665662343485334,
 'cnlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.17865975536217787,
 'cblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2211363687940211,
 'rcalm03_Ground_portion_of_LM-LAM-002': 0.09669401081425906,
 'calm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.21355011410002714,
 'calm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.19579826831296507,
 'calm35_Breccia__Lunar_Feldspathic_Breccia': 0.2596924006141492,
 'c1lm34_Breccia__Lunar_Feldspathic_Breccia': 0.2507492135722991,
 'interpolated_data': 0.21625214235890244,
 'cblm03_Ground_portion_of_LM-LAM-002': 0.0967870105936377,
 'c10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.254157648694993,
 'c1lm51b_Lunar_meteorite__feldspathic_fragmental_breccia':
0.152727608829985,
 'chlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24547976992089954,
 's10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25223018977441714,
 'bir1lm045_unbrecciated_lunar_meteorite__coarse_texture':
0.2438167181037139,
 'calm03_Ground_portion_of_LM-LAM-002': 0.0962091955449693,
 'srlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2466093268834993,
 'cdlm35_Breccia__Lunar_Feldspathic_Breccia': 0.2514285069163198,

'camt313_Igneous__Olivine_Gabbro__Purchased_from_catchafallingstar.com
': 0.10053545033650797,
 'c7mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.3796803425023198,
 'c2lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.283624205885601,
```

'celm34_Breccia__Lunar_Feldspathic_Breccia': 0.256293101550118,
'c10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24874081534038098,
'c2lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
0.1410599264557316,
'calmcr_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.0841094775032077,
'cblm13_Mostly_black_side_opposite_of_(A).': 0.24976639074903403,
'calmn1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.203955794615469,
'c1lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.2868455812445023,
'c1lm39_Igneous__Lunar_Unbrecciated_Basalt_': 0.3048130559262863,
'comb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.11835318662241029,
'calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.07368270627196831,
'c4mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.2352106834731243,
'cslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25067457831595524,
'celm35_Breccia__Lunar_Feldspathic_Breccia': 0.27426039923158746,
'sblm03_Ground_portion_of_LM-LAM-002': 0.0982997135056761,
'cdlm13_(D)_Bottom_black_flat_with_brown_spot.': 0.16611577752426632,
'c1lm38_Breccia__Lunar_Basaltic_Breccia': 0.21625214235890244,
'cflm35_Breccia__Lunar_Feldspathic_Breccia': 0.25482686741269006,
'solm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.26452433020261823,
'calmn2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.20627739666993147,
'calmca_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.07847614707398592,
'c1lm37_Breccia__Lunar_Basaltic_Breccia': 0.18889514282483064,
'shlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24538751075770202,
'cqmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.19818160441605825}

sam_noise_25

{'c2mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.26030971155621885,
'bir1lm046_unbrecciated_lunar_meteorite__coarse_texture':
0.18223921861704012,
'c1lm36_Igneous__Lunar_Unbrecciated_Basalt_': 0.26527061175914574,
'cclm35_Breccia__Lunar_Feldspathic_Breccia': 0.25695663984383665,

```
 'c10l01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2422568854538411,
 'c3mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.406512747541019,
 'colm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25942278785096595,
 'c12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.22949414679663155,
 'c6mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.3005600051670467,
 'bir1lm048_Mare_basalt': 0.159544747780975,

'cama01_Anomalous_(ACANOM)__Antarctic_meteorite_of_presumed_lunar_orig
in': 0.20872258111969536,
 'calm34_Breccia__Lunar_Feldspathic_Breccia': 0.2868064437826109,
 'cflm34_Breccia__Lunar_Feldspathic_Breccia': 0.2667984777187317,
 'cpmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.0875721374532911,
 'c1lm50b_0.148_g': 0.21925110542620332,
 'cclm34_Breccia__Lunar_Feldspathic_Breccia': 0.20879474866913197,
 'cblm34_Breccia__Lunar_Feldspathic_Breccia': 0.22422444414104117,
 'bir1lm053_Breccia__Lunar_Feldspathic_Breccia': 0.09525995820292574,
 'celm13_(E)_White_end.': 0.28238763203046124,
 'bir1lm044_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.21766369918511325,
 'c1lm52b_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.1913275156299958,
 'c1lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
0.1415840211068108,
 'c1lm51_Lunar_meteorite__feldspathic_fragmental_breccia':
0.3042338458866851,
 's12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.22644959661218536,
 'c1lm52c_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.12028549957754364,
 'cclm13_Top_whitish.': 0.2394415531508529,
 'cblm35_Breccia__Lunar_Feldspathic_Breccia': 0.25921808073131647,
 'c1lm50c_0.148_g': 0.17202026224707673,
 'sblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.22853543802291862,
 'c2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.27250223888962927,
 's2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25239268759399663,
 'bir1lm047_Mare_basalt': 0.18702682969064277,
 'sslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
```

```
0.24761021655214935,
 'c1lm50_0.148_g': 0.2764838783219293,
 'c1lm51c_Lunar_meteorite__feldspathic_fragmental_breccia':
0.15436365423716888,
 'bir1lm054_Breccia__Lunar_Feldspathic_Breccia': 0.08830315006885042,
 's10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25833912292902694,
 'c8mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.3733143256333454,
 'c9mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.18309295699380043,
 'c1lm52_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.22993038710834843,
 'cdlm34_Breccia__Lunar_Feldspathic_Breccia': 0.32016779687731095,
 'crlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2443449376287841,
 'cnmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.12678722487950297,
 'cnlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.17851639315879242,
 'cblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2224945312917053,
 'rcalm03_Ground_portion_of_LM-LAM-002': 0.0961594118110531,
 'calm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.21742178504540274,
 'calm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.1956078400564204,
 'calm35_Breccia__Lunar_Feldspathic_Breccia': 0.26086445725626617,
 'c1lm34_Breccia__Lunar_Feldspathic_Breccia': 0.25423633163255877,
 'interpolated_data': 0.22021149909843551,
 'cblm03_Ground_portion_of_LM-LAM-002': 0.09737109965993751,
 'c10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25315986331590107,
 'c1lm51b_Lunar_meteorite__feldspathic_fragmental_breccia':
0.15530496821501047,
 'chlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2431561750276168,
 's10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25810761658028714,
 'bir1lm045_unbrecciated_lunar_meteorite__coarse_texture':
0.23996213006180062,
 'calm03_Ground_portion_of_LM-LAM-002': 0.09492747110033006,
 'srlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2472320256959818,
 'cdlm35_Breccia__Lunar_Feldspathic_Breccia': 0.2519813925290549,
```

```
'camt313_Igneous__Olivine_Gabbro__Purchased_from_catchafallingstar.com
': 0.10033165404327107,
 'c7mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.37833904757759934,
 'c2lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.2843275038371477,
 'celm34_Breccia__Lunar_Feldspathic_Breccia': 0.2587461620882935,
 'c10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24383550684184943,
 'c2lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
0.1417873578779756,
 'calmcr_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.0792578699779261,
 'cblm13_Mostly_black_side_opposite_of_(A).': 0.2497122429429989,
 'calmn1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2058895392125661,
 'c1lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.2864741499663774,
 'c1lm39_Igneous__Lunar_Unbrecciated_Basalt_': 0.3120645302392668,
 'comb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.11879563790604818,
 'calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.07690445215674684,
 'c4mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.23641525689661133,
 'cslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2514387003876467,
 'celm35_Breccia__Lunar_Feldspathic_Breccia': 0.2792214352914283,
 'sblm03_Ground_portion_of_LM-LAM-002': 0.09607549801301037,
 'cdlm13_(D)_Bottom_black_flat_with_brown_spot.': 0.16675086512472023,
 'c1lm38_Breccia__Lunar_Basaltic_Breccia': 0.21867062338058807,
 'cflm35_Breccia__Lunar_Feldspathic_Breccia': 0.252371358666767,
 'solm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.26319678338563046,
 'calmn2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.20592997346956346,
 'calmca_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.08369350897465955,
 'c1lm37_Breccia__Lunar_Basaltic_Breccia': 0.1900201433415223,
 'shlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24608818892247913,
 'cqmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.19861746724248416}

sam_noise_5
```

```
{'c2mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.2561143015091069,
 'bir1lm046_unbrecciated_lunar_meteorite__coarse_texture':
0.18872816376522894,
 'c1lm36_Igneous__Lunar_Unbrecciated_Basalt_': 0.256386834549953,
 'cclm35_Breccia__Lunar_Feldspathic_Breccia': 0.2725709505485384,
 'c10l01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25091180391199697,
 'c3mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.40586251323672856,
 'colm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2606540431799786,
 'c12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.23073872030460588,
 'c6mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.301161985183611,
 'bir1lm048_Mare_basalt': 0.1637584225067206,

'cama01_Anomalous_(ACANOM)__Antarctic_meteorite_of_presumed_lunar_orig
in': 0.19827181583580203,
 'calm34_Breccia__Lunar_Feldspathic_Breccia': 0.2888559663841911,
 'cflm34_Breccia__Lunar_Feldspathic_Breccia': 0.26856965951010864,
 'cpmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.08370493770359685,
 'c1lm50b_0.148_g': 0.22084084141082727,
 'cclm34_Breccia__Lunar_Feldspathic_Breccia': 0.2039418741892438,
 'cblm34_Breccia__Lunar_Feldspathic_Breccia': 0.228631773686972,
 'bir1lm053_Breccia__Lunar_Feldspathic_Breccia': 0.0983895112299535,
 'celm13_(E)_White_end.': 0.28720291778344614,
 'bir1lm044_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.21703585577207832,
 'c1lm52b_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.18767487770270846,
 'c1lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
0.1409898905469343,
 'c1lm51_Lunar_meteorite__feldspathic_fragmental_breccia':
0.3051163123052569,
 's12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.22880721998085018,
 'c1lm52c_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.1259148649039953,
 'cclm13_Top_whitish.': 0.2409226661210096,
 'cblm35_Breccia__Lunar_Feldspathic_Breccia': 0.24167493205890198,
 'c1lm50c_0.148_g': 0.17092213885470683,
 'sblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.23375447163476126,
```

```
 'c2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2626587351337689,
 's2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.27362969510090207,
 'bir1lm047_Mare_basalt': 0.19855217422815466,
 'sslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24541765560853454,
 'c1lm50_0.148_g': 0.27836313733584134,
 'c1lm51c_Lunar_meteorite__feldspathic_fragmental_breccia':
0.15607270068740078,
 'bir1lm054_Breccia__Lunar_Feldspathic_Breccia': 0.08439629990756731,
 's10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2609581674025582,
 'c8mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.3820479011017947,
 'c9mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.17157712213146636,
 'c1lm52_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
0.22989027463098952,
 'cdlm34_Breccia__Lunar_Feldspathic_Breccia': 0.3202648642233306,
 'crlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.24914612300062206,
 'cnmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.12658175927112933,
 'cnlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.1840669290393874,
 'cblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.22673956223564684,
 'rcalm03_Ground_portion_of_LM-LAM-002': 0.10038904598826273,
 'calm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2095609882145272,
 'calm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.19801979937196665,
 'calm35_Breccia__Lunar_Feldspathic_Breccia': 0.25169322680032835,
 'c1lm34_Breccia__Lunar_Feldspathic_Breccia': 0.2501358071618887,
 'interpolated_data': 0.21583148164948973,
 'cblm03_Ground_portion_of_LM-LAM-002': 0.10052544343169535,
 'c10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2622295923829096,
 'c1lm51b_Lunar_meteorite__feldspathic_fragmental_breccia':
0.15064559019445026,
 'chlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.26451884001606873,
 's10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25426900435558003,
 'bir1lm045_unbrecciated_lunar_meteorite__coarse_texture':
0.248390634775454,
```

```
 'calm03_Ground_portion_of_LM-LAM-002': 0.09709051915313235,
 'srlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2508625216795921,
 'cdlm35_Breccia__Lunar_Feldspathic_Breccia': 0.24985590381714465,

'camt313_Igneous__Olivine_Gabbro__Purchased_from_catchafallingstar.com
': 0.10105781736071358,
 'c7mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.3962519149759978,
 'c2lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.28499842026817795,
 'celm34_Breccia__Lunar_Feldspathic_Breccia': 0.24683892084796513,
 'c10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.25187554988465755,
 'c2lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
0.14084539192545584,
 'calmcr_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.08290146748619312,
 'cblm13_Mostly_black_side_opposite_of_(A).': 0.25074317473343366,
 'calmn1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.20303879713983167,
 'c1lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
0.2891914594975031,
 'c1lm39_Igneous__Lunar_Unbrecciated_Basalt_': 0.30880093567947847,
 'comb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.11812869057851012,
 'calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.07330948389031006,
 'c4mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.24662111901546566,
 'cslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2488295649816096,
 'celm35_Breccia__Lunar_Feldspathic_Breccia': 0.2915182267516025,
 'sblm03_Ground_portion_of_LM-LAM-002': 0.09801175479656415,
 'cdlm13_(D)_Bottom_black_flat_with_brown_spot.': 0.16353791684189853,
 'c1lm38_Breccia__Lunar_Basaltic_Breccia': 0.22200663337226684,
 'cflm35_Breccia__Lunar_Feldspathic_Breccia': 0.256167877449616,
 'solm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2707548250875124,
 'calmn2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2052934507289216,
 'calmca_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.08358205318628094,
 'c1lm37_Breccia__Lunar_Basaltic_Breccia': 0.1950992105875568,
 'shlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
0.2352037934766225,
 'cqmb29_Igneous__Gabbroic__A-
```

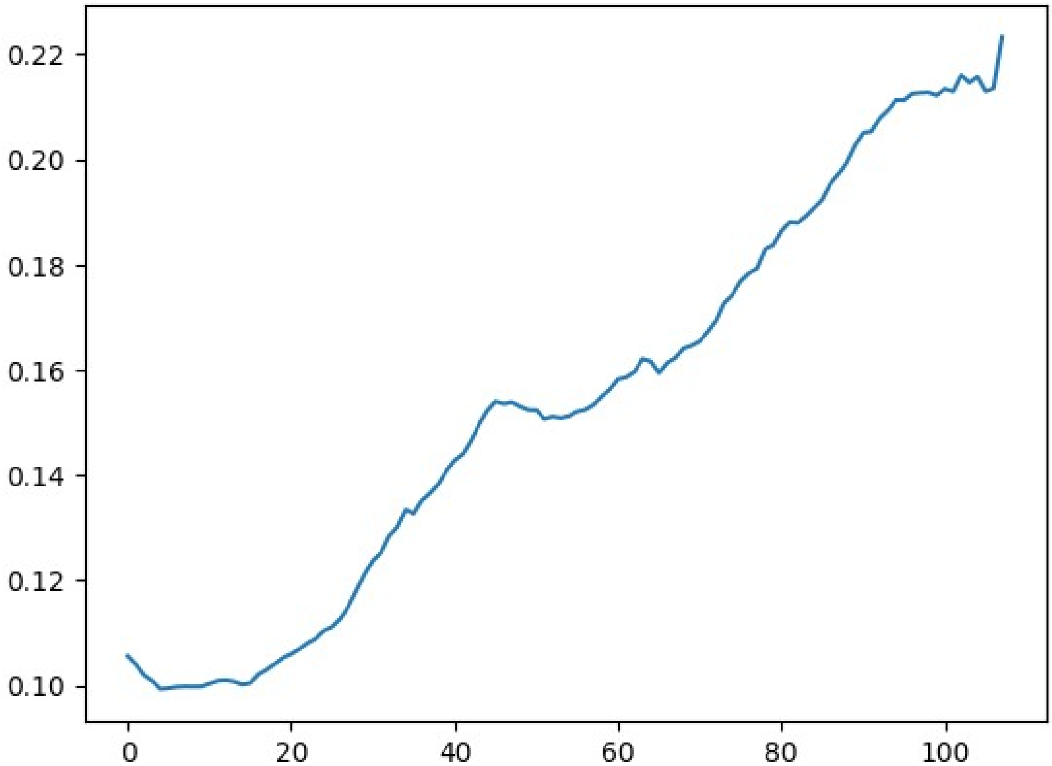
```
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
0.19628999973675762}
```

```
temp = min(sam.values())
res = [key for key in sam if sam[key] == temp]

temp25 = min(sam_noise_25.values())
res25 = [key for key in sam_noise_25 if sam_noise_25[key] == temp25]

temp5 = min(sam_noise_5.values())
res5 = [key for key in sam_noise_5 if sam_noise_5[key] == temp5]

print(temp, res)
print(temp25, res25)
print(temp5, res5)
```

```
0.07368270627196831
['calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin']
0.07690445215674684
['calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin']
0.07330948389031006
['calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin']
```

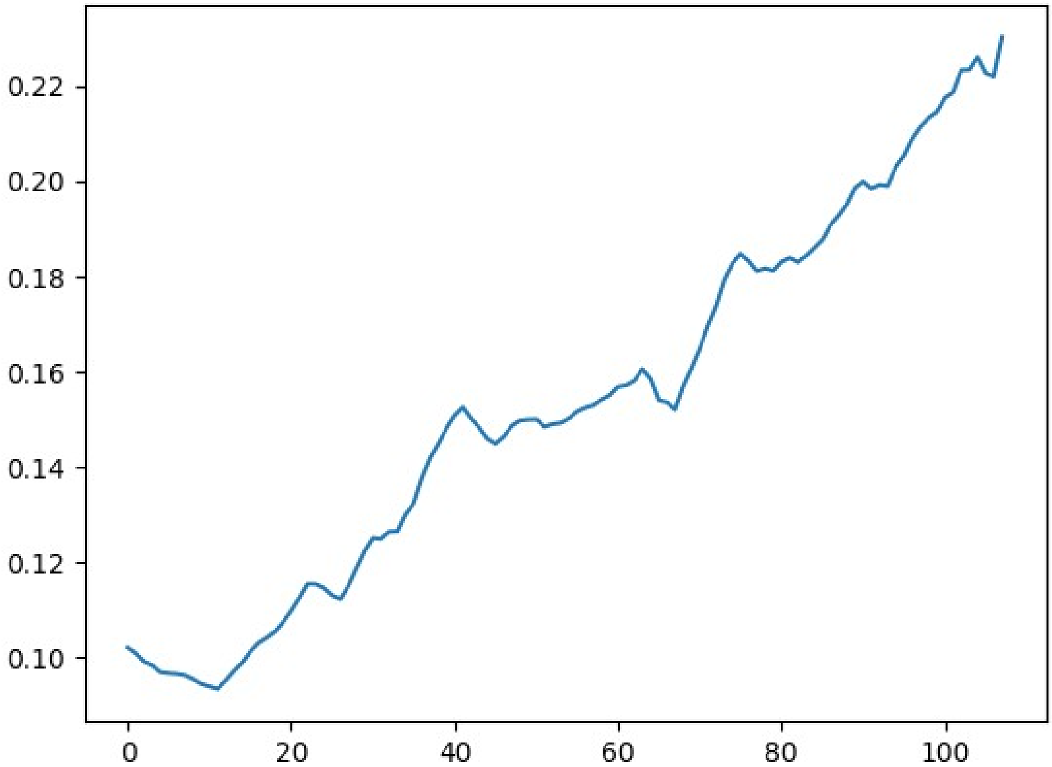
```
plt.plot(reflectance_vectors_noise_25['calmc2_Antarctic_Meteorite_of_P
resumed_Lunar_Origin'])
```

```
[<matplotlib.lines.Line2D at 0x7fc82f94ee30>]
```



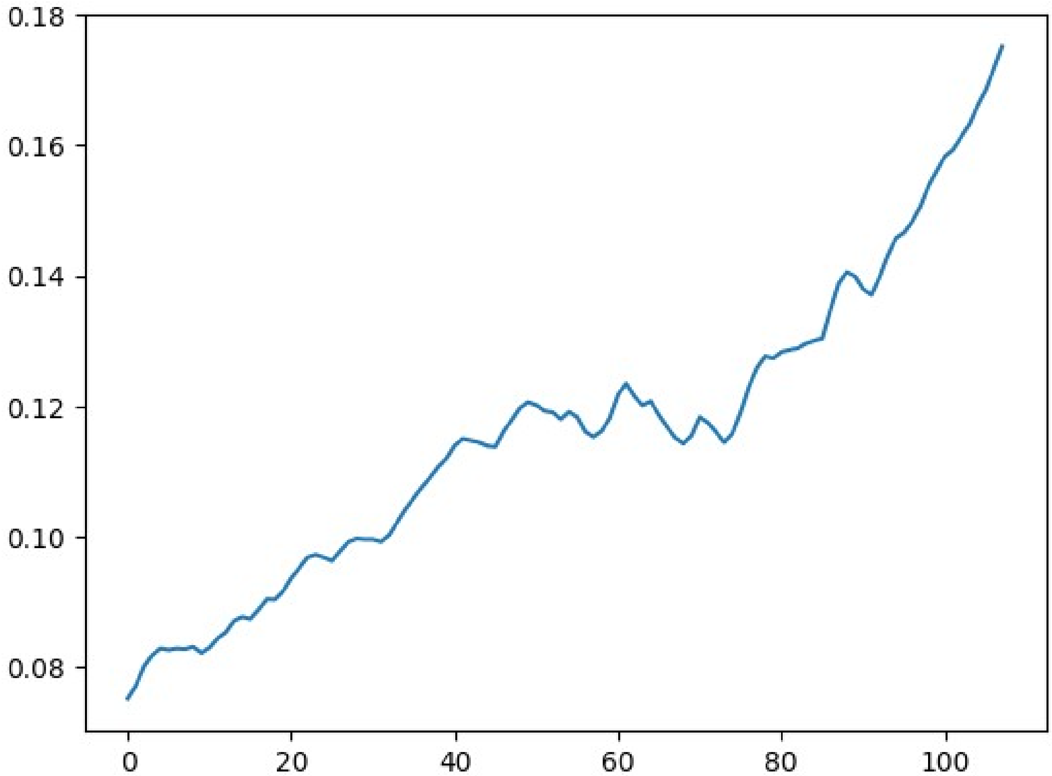

```
plt.plot(reflectance_vectors_noise_5['calmc2_Antarctic_Meteorite_of_Pr
esumed_Lunar_Origin'])
```

```
[<matplotlib.lines.Line2D at 0x7fc82f9a8bb0>]
```




```
plt.plot(denoised_reflectance_image[7:115, 1000, 200])
```

```
[<matplotlib.lines.Line2D at 0x7fc82f6df1f0>]
```



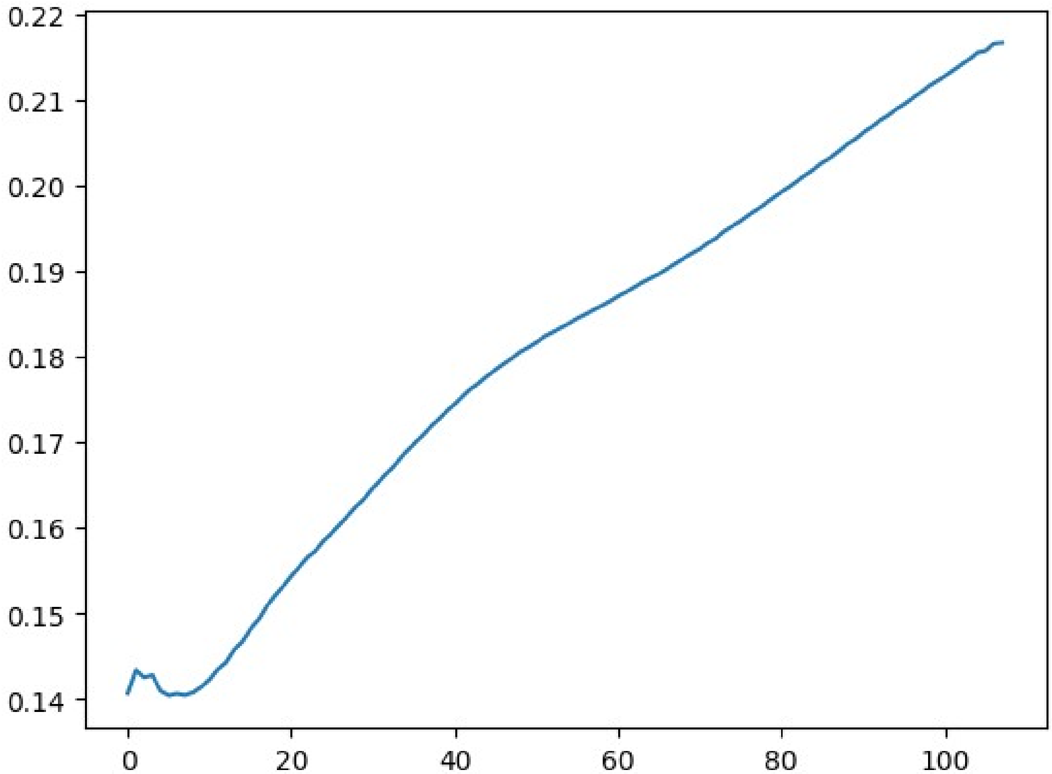

```
min_sam_key = min(reflectance_vectors.keys(), key=lambda k:
sam[list(reflectance_vectors.keys()).index(k)])
min_sam_key

---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call
last)
Cell In[28], line 1
----> 1 min_sam_key = min(reflectance_vectors.keys(), key=lambda k:
sam[list(reflectance_vectors.keys()).index(k)])
      2 min_sam_key

Cell In[28], line 1, in <lambda>(k)
----> 1 min_sam_key = min(reflectance_vectors.keys(), key=lambda k:
sam[list(reflectance_vectors.keys()).index(k)])
      2 min_sam_key

KeyError: 0

# Create a list of tuples with (key, sam_value)
sam_key_pairs = list(reflectance_vectors.keys())

# Sort the list based on the SAM values
sorted_sam_key_pairs = sorted(sam_key_pairs, key=lambda k:
sam[sam_key_pairs.index(k)])
```

```
# Extract the minimum 5 pairs
min_5_sam_key_pairs = sorted_sam_key_pairs[:5]

# Get the corresponding SAM values for these keys
min_5_sam_values = [sam[sam_key_pairs.index(k)] for k in
min_5_sam_key_pairs]

# Combine them into a list of tuples (key, sam_value)
min_5_results = list(zip(min_5_sam_key_pairs, min_5_sam_values))

min_5_results
```

```
[('calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin',
0.07368270627196831),
 ('calmca_Antarctic_Meteorite_of_Presumed_Lunar_Origin',
0.07847614707398592),
 ('bir1lm054_Breccia__Lunar_Feldspathic_Breccia', 0.0835425883809933),
 ('calmcr_Antarctic_Meteorite_of_Presumed_Lunar_Origin',
0.0841094775032077),
 ('cpmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite',
  0.08428332167762563)]
```

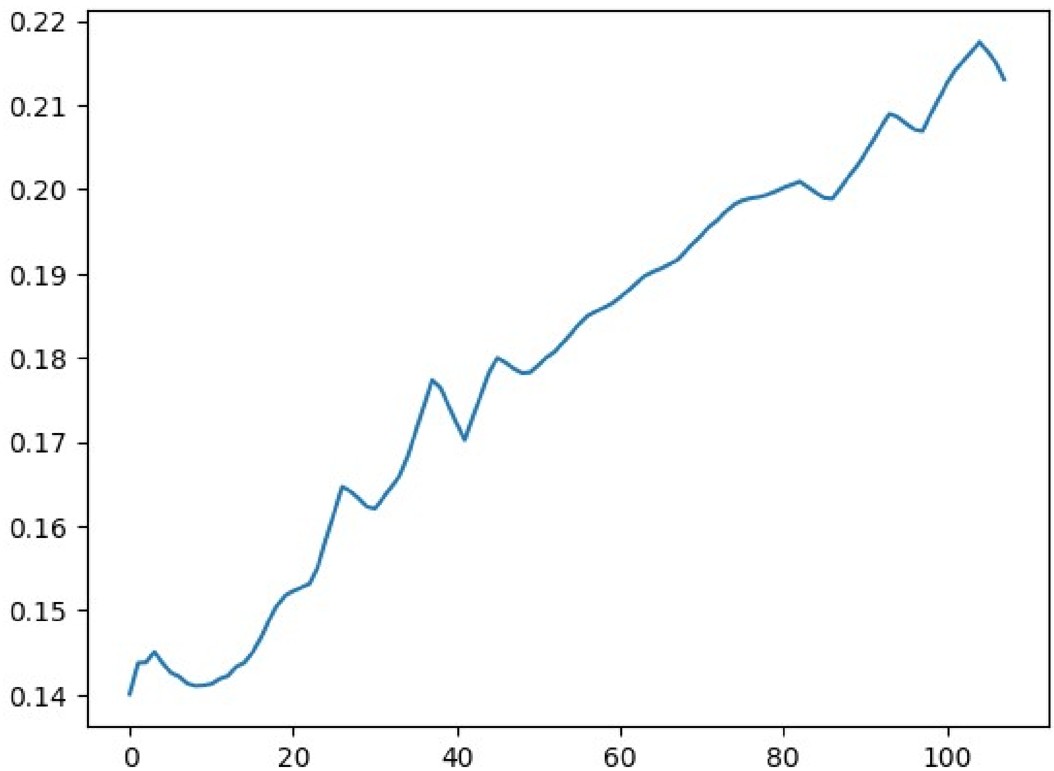
```
plt.plot(reflectance_vectors['bir1lm054_Breccia__Lunar_Feldspathic_Bre
ccia'])
```

```
[<matplotlib.lines.Line2D at 0x7ca1c2775810>]
```



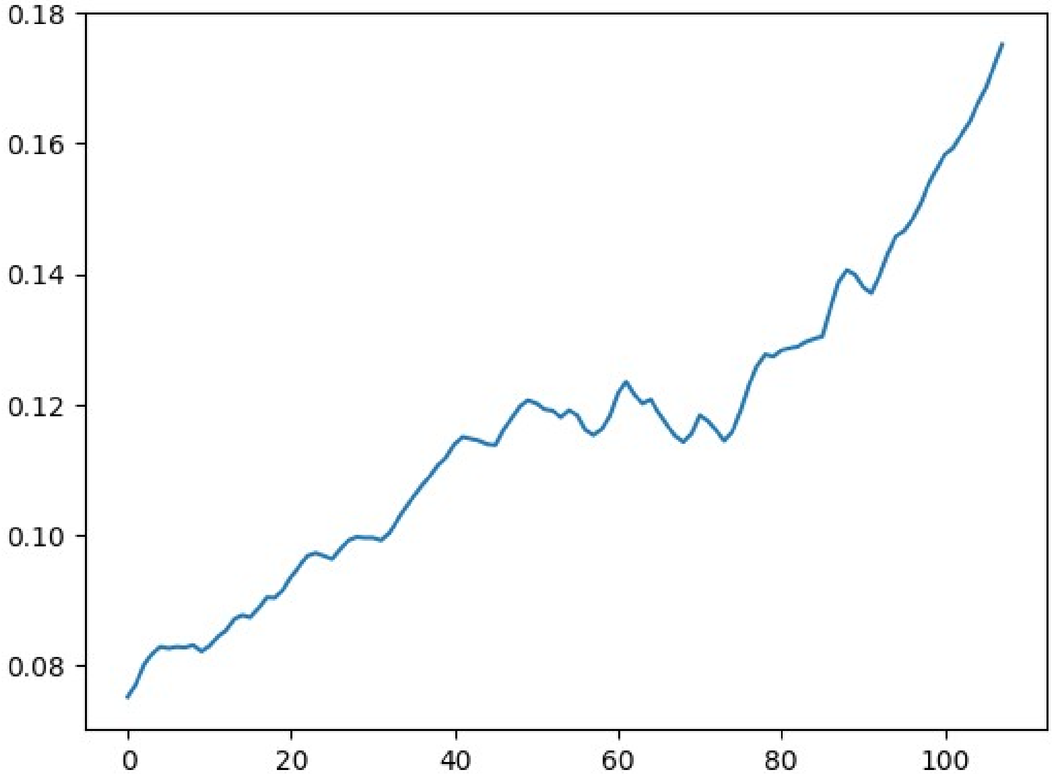

```
import numpy as np
import matplotlib.pyplot as plt


# plt.plot(reflectance_noisy)
```

```
[<matplotlib.lines.Line2D at 0x7ca1c19b1600>]
```

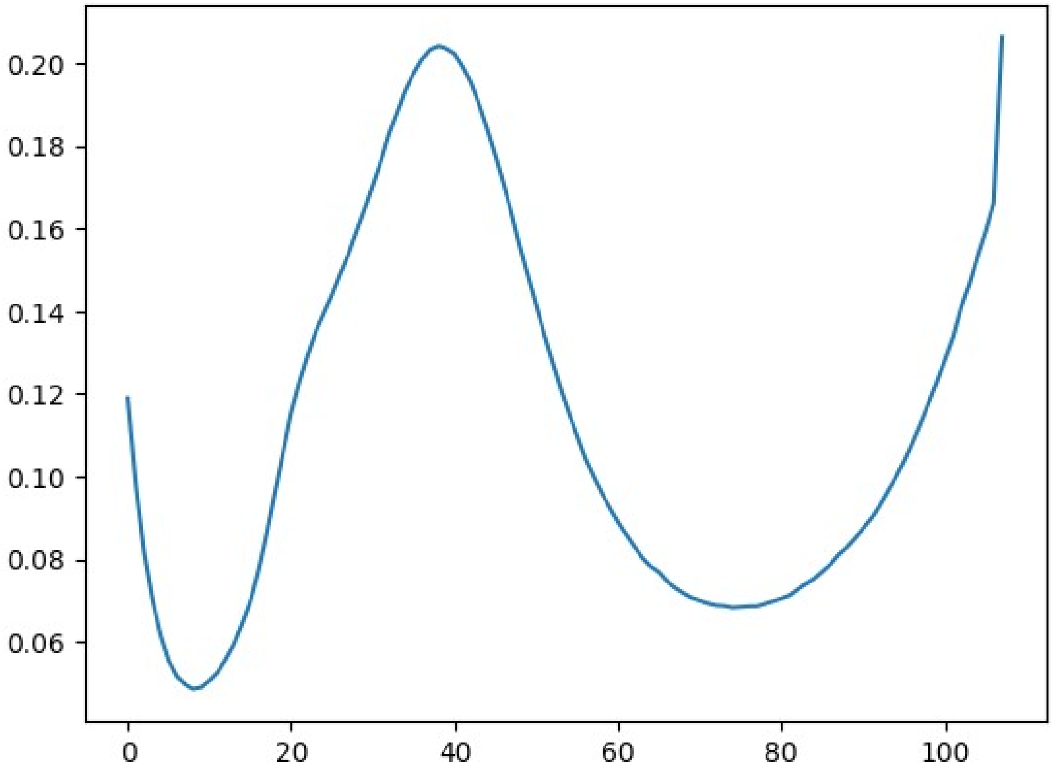



```
import plotly.graph_objs as go
import plotly.offline as pyo

# Create a plotly figure
fig = go.Figure()

# Loop over each key in the dictionary and add a line plot for each
for key, vector in reflectance_vectors.items():
    fig.add_trace(go.Scatter(y=vector, mode='lines', name=key))

# Customize layout
fig.update_layout(
    title="Reflectance Vectors",
    xaxis_title="Wavelength",
    yaxis_title="Reflectance",
    showlegend=True
)

# Show the figure
pyo.plot(fig)

'temp-plot.html'

import matplotlib.pyplot as plt

plt.plot(denoised_reflectance_image[7:115, 1000, 200])
```

```
[<matplotlib.lines.Line2D at 0x7ca1c2757c40>]
```




```
plt.plot(reflectance_vectors[min_sam_key])

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[3], line 1
----> 1 plt.plot(reflectance_vectors[min_sam_key])

NameError: name 'plt' is not defined

max_sam_key = max(reflectance_vectors.keys(), key=lambda k: sam[list(reflectance_vectors.keys()).index(k)])
max_sam_key

'c3mb29_Silicate_(Ino)__Pyroxene__A-881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite'

plt.plot(reflectance_vectors[max_sam_key])

[<matplotlib.lines.Line2D at 0x7ca1c2bb96c0>]
```




```
SAM = np.zeros((5, 5, len(reflectance_vectors)))

for i in range(10000, 10005):
    for j in range(100, 105):
        for k, v in enumerate(reflectance_vectors.values()):
            SAM[i - 10000, j - 100, k] =
spectral_angle_mapper(reflectance_image[7:115, i, j], v)

SAM

array([[[0.31590569, 0.23851635, 0.31192783, ..., 0.25022716,
         0.30881818, 0.26119096],
        [0.31368341, 0.23613191, 0.30972014, ..., 0.24766726,
         0.30599722, 0.25830321],
        [0.31094703, 0.23312169, 0.30692437, ..., 0.24470094,
         0.30304179, 0.25517423],
        [0.31380457, 0.23615154, 0.3097159 , ..., 0.24797507,
         0.30667356, 0.2588868 ],
        [0.31609226, 0.23866319, 0.31208122, ..., 0.25059408,
         0.30908073, 0.26145042]],

       [[0.31564148, 0.2381912 , 0.31171761, ..., 0.24990554,
         0.30863233, 0.26103509],
        [0.31396327, 0.23636349, 0.31002019, ..., 0.24794527,
         0.30634838, 0.25864705],
        [0.31232972, 0.23477165, 0.30844325, ..., 0.24643688,
```

```
         0.30507962, 0.25746248],
        [0.31536437, 0.23787968, 0.31139914, ..., 0.24968947,
         0.30837015, 0.26073931],
        [0.31719415, 0.23980885, 0.31322303, ..., 0.25141122,
         0.31012204, 0.26252107]],

       [[0.31502401, 0.23762248, 0.31109775, ..., 0.24917595,
         0.30787092, 0.26030071],
        [0.31444325, 0.23690559, 0.31052558, ..., 0.24859732,
         0.30701469, 0.25939006],
        [0.31249625, 0.23488729, 0.30867667, ..., 0.24607814,
         0.30452422, 0.25687656],
        [0.3138525 , 0.23636832, 0.3098477 , ..., 0.2480105 ,
         0.30678987, 0.25913594],
        [0.31668406, 0.23939328, 0.31280075, ..., 0.25075552,
         0.30936261, 0.26186162]],

       [[0.31396467, 0.23663725, 0.31001644, ..., 0.2479332 ,
         0.30678823, 0.25923522],
        [0.31137054, 0.23380745, 0.30747036, ..., 0.24484579,
         0.30338265, 0.25571149],
        [0.3117178 , 0.2342329 , 0.30792912, ..., 0.24540394,
         0.30386595, 0.25634032],
        [0.31231407, 0.23474834, 0.3084842 , ..., 0.24608542,
         0.30453866, 0.25694782],
        [0.31374681, 0.23626264, 0.30987825, ..., 0.24768297,
         0.30630178, 0.25871052]],

       [[0.31286703, 0.23524388, 0.30897695, ..., 0.24659383,
         0.30492894, 0.25720631],
        [0.31442462, 0.23684185, 0.31059918, ..., 0.24810221,
         0.30659295, 0.25895326],
        [0.31630784, 0.23890018, 0.31239511, ..., 0.25014497,
         0.30912223, 0.26156166],
        [0.31591009, 0.23852879, 0.31204985, ..., 0.24957057,
         0.30838635, 0.26082664],
        [0.31611059, 0.23877366, 0.3121361 , ..., 0.25021343,
         0.3093591 , 0.26178088]]]])
```

```
max_indices = np.argmax(SAM, axis=2)
min_indices = np.argmin(SAM, axis=2)

max_indices
```

```
array([[5, 5, 5, 5, 5],
       [5, 5, 5, 5, 5],
       [5, 5, 5, 5, 5],
       [5, 5, 5, 5, 5],
       [5, 5, 5, 5, 5]])
```

```
min_indices
```

```
array([[72, 72, 72, 72, 72],
       [72, 72, 72, 72, 72],
       [72, 72, 72, 72, 72],
       [72, 72, 72, 72, 72],
       [72, 72, 72, 72, 72]])
```

```
plt.plot(reflectance_image[7:115, 0, 0])

plt.plot(reflectance_vectors[list(reflectance_vectors.keys())[72]])

list(reflectance_vectors.keys())[5]

plt.plot(reflectance_vectors[list(reflectance_vectors.keys())[5]])

min_sam = np.min(SAM, axis=2)
min_sam

min_value = np.min(SAM)

min_coords = np.unravel_index(np.argmin(SAM), SAM.shape)

min_value

min_coords

plt.plot(reflectance_image[7:115, 153, 248])

plt.plot(reflectance_vectors[list(reflectance_vectors.keys())[76]])

import torch

# Convert the reflectance image and reflectance vectors to PyTorch
tensors
reflectance_image_tensor =
torch.tensor(reflectance_image[7:115, :, :],
dtype=torch.float32).cuda()
reflectance_vectors_tensor = torch.tensor([v for v in
reflectance_vectors.values()], dtype=torch.float32).cuda()

# Initialize a 3D tensor for SAM with the same spatial dimensions as
the reflectance_image
SAM = torch.zeros((reflectance_image.shape[1],
reflectance_image.shape[2], len(reflectance_vectors))).cuda()

# Define the spectral_angle_mapper function for GPU usage
def spectral_angle_mapper_gpu(pixel_vector, ref_vector):
    dot_product = torch.dot(pixel_vector, ref_vector)
    norm_pixel = torch.norm(pixel_vector)
    norm_ref = torch.norm(ref_vector)
    angle = torch.acos(dot_product / (norm_pixel * norm_ref))
```

```
    return angle

# Calculate the SAM value for each pixel and store it in the SAM
tensor
for i in range(reflectance_image.shape[1]):
    for j in range(reflectance_image.shape[2]):
        pixel_vector = reflectance_image_tensor[:, i, j]  # Get the
spectral vector for pixel (i, j)
        for k, ref_vector in enumerate(reflectance_vectors_tensor):
            SAM[i, j, k] = spectral_angle_mapper_gpu(pixel_vector,
ref_vector)

    if i % 1000 == 0:
        print(f'Processing row {i}/{reflectance_image.shape[1]}')

# If needed, move the SAM tensor back to CPU and convert it to a NumPy
array
SAM_cpu = SAM.cpu().numpy()

SAM.shape

SAM

min_sam_key = min(reflectance_vectors.keys(), key=lambda k:
SAM[list(reflectance_vectors.keys()).index(k)])
min_sam_key

# SAM = spectral_angle_mapper(reflectance_image[7:115, 0, 0],
interpolated_reflectance)

# SAM
```

CPRSM

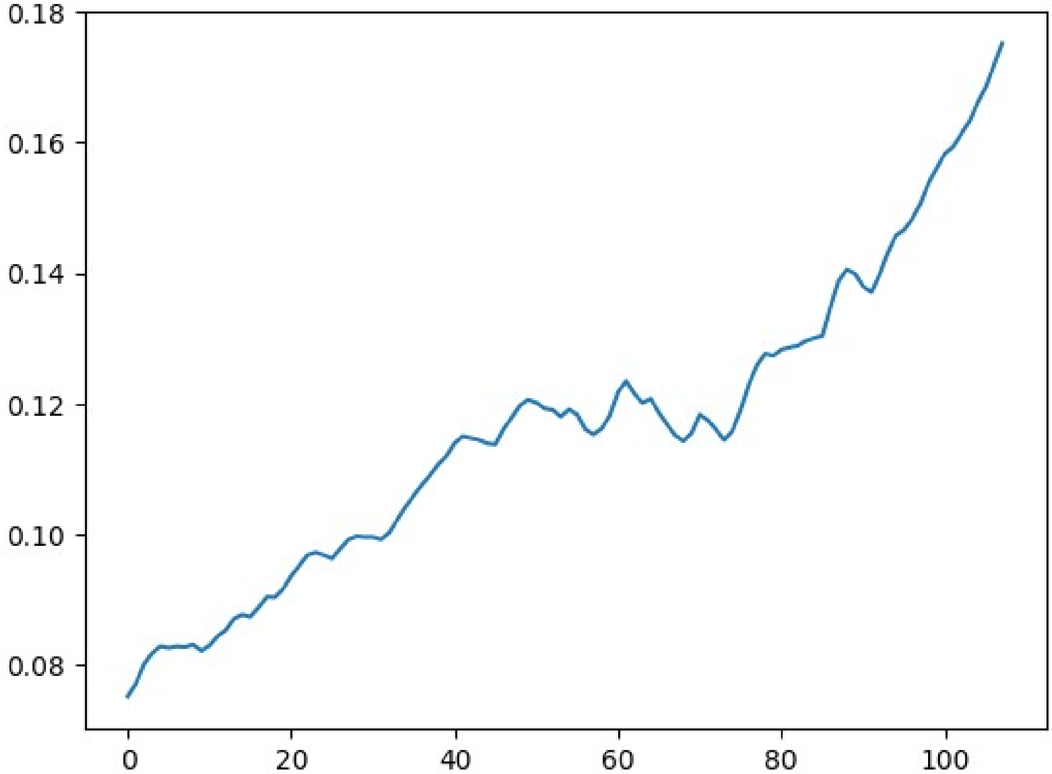
```
import numpy as np

def CPRMS(pixel_vector, reference_vector):

    mean_pixel = np.mean(pixel_vector)
    mean_reference = np.mean(reference_vector)

    centered_pixel = pixel_vector - mean_pixel
    centered_reference = reference_vector - mean_reference

    squared_diff = (centered_pixel - centered_reference) ** 2

    val = np.sqrt(np.mean(squared_diff))

    return val
```

```python
cprms_values = {}
cprms_noise_25 = {}
cprms_noise_5 = {}

for key1, key2, key3 in zip(reflectance_vectors.keys(),
                            reflectance_vectors_noise_25.keys(),
                            reflectance_vectors_noise_5.keys()):
    ref_vec = denoised_reflectance_image[7:115, 1000, 200]

    cprms_values[key1] = CPRMS(ref_vec, reflectance_vectors[key1])
    cprms_noise_25[key2] = CPRMS(ref_vec,
reflectance_vectors_noise_25[key2])
    cprms_noise_5[key3] = CPRMS(ref_vec,
reflectance_vectors_noise_5[key3])

temp = min(cprms_values.values())
res = [key for key in cprms_values if cprms_values[key] == temp]

temp25 = min(cprms_noise_25.values())
res25 = [key for key in cprms_noise_25 if cprms_noise_25[key] ==
temp25]

temp5 = min(cprms_noise_5.values())
res5 = [key for key in cprms_noise_5 if cprms_noise_5[key] == temp5]

print(temp, res)
print(temp25, res25)
print(temp5, res5)
```

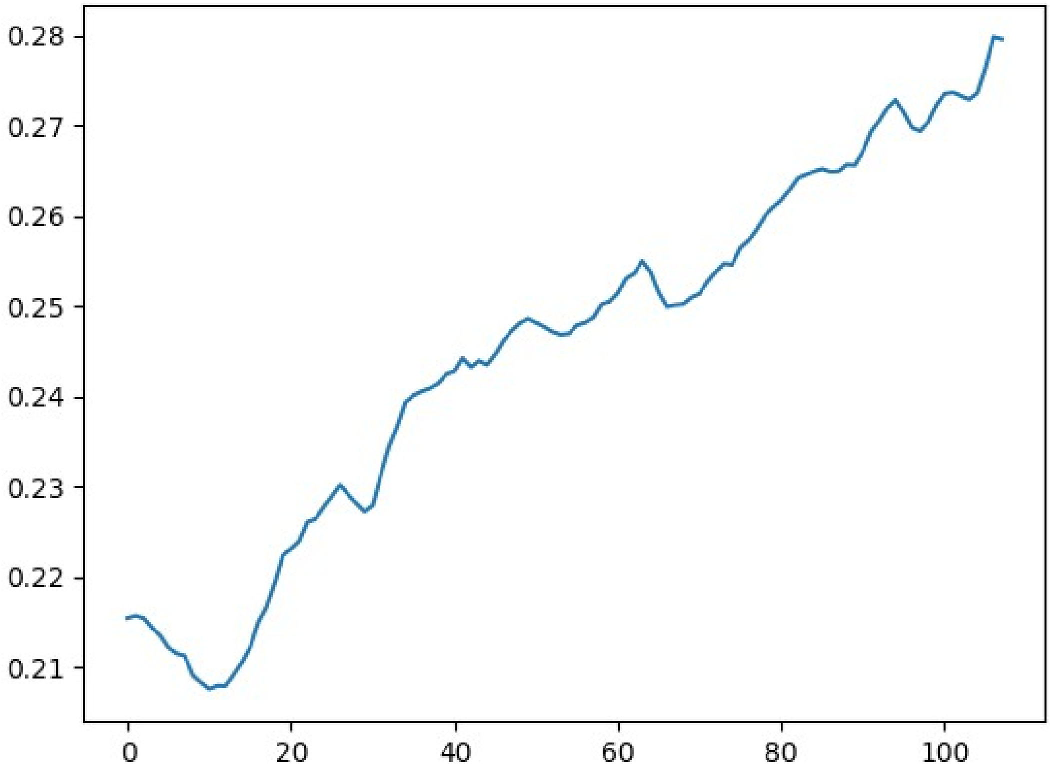
```
0.006582425490993638
['c1lm52c_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g']
0.006965808944762407
['c1lm52c_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g']
0.008180757674315827
['c1lm52c_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g']
```

```python
plt.plot(ref_vec)
```

```
[<matplotlib.lines.Line2D at 0x7fc82f1bb2b0>]
```




```
plt.plot(reflectance_vectors_noise_25['c1lm52c_Breccia__Lunar_Feldspat
hic_Regolith_Breccia__0.97_g'])

[<matplotlib.lines.Line2D at 0x7fc82ee14e20>]
```




```
cprms = []
for v in reflectance_vectors.values():
    cprms.append(spectral_angle_mapper(reflectance_image[7:115, 0, 0],
v))

cprms

min_cprms_key = min(reflectance_vectors.keys(), key=lambda k:
cprms[list(reflectance_vectors.keys()).index(k)])
min_cprms_key

import matplotlib.pyplot as plt

plt.plot(reflectance_image[7:115, 0, 0])

plt.plot(reflectance_vectors[min_cprms_key])

plt.plot(reflectance_vectors[max_cprms_key])
```

Torch Dataset

```
import os
import numpy as np
import torchdata.datapipes as dp
from torch.utils.data import DataLoader
```

```
class SlidingWindowDataPipe(dp.iter.IterDataPipe):
    def __init__(self, image_paths, window_size, step_size):
        super().__init__()
        self.image_paths = image_paths
        self.window_size = window_size
        self.step_size = step_size

    def __iter__(self):
        for image_path in self.image_paths:
            shapes =
SlidingWindowDataPipe.get_image_meta_data(image_path)
            for shape in shapes:
                _,img_height, img_width = shape
                # Sliding window
                for i in range(0, img_height - self.window_size + 1,
self.step_size):
                    if i + self.window_size > img_height:
                        continue
                    patch =
SlidingWindowDataPipe.get_partial_image_from_height(image_path,i,self.
window_size)
                    yield patch
    @staticmethod
    def _read_partial_data_of_given_height(qub_path,image_height,
image_width,row = 0, height = 250):
        image = []
        data_count = image_width * height
        channel_size = image_height * image_width
        with open(qub_path,'rb') as f:
            for channel_idx in range(256):
                offset = channel_idx * channel_size * 4 + row *
image_width  # float32 has 4 bytes
                f.seek(offset)
                channel_data = np.fromfile(f, dtype=np.float32,
count=data_count)
                image.append(channel_data.reshape((1,height,
image_width)))
        return np.vstack(image)
    @staticmethod
    def get_partial_image_from_height(base_path, row, height):
        xml_files = utils.find_xml_files(base_path)
        image_files = utils.find_qub_files(base_path)
        shapes = [utils.extract_sequence_numbers(xml_file) for
xml_file in xml_files]
        images =
[utils._read_partial_data_of_given_height(qub_path,shape[1], shape[2],
row, height) for qub_path,shape in zip(image_files,shapes)]
        return images
```

```python
    @staticmethod
    def get_image_meta_data(base_path):
        xml_files = utils.find_xml_files(base_path)
        shapes = [utils.extract_sequence_numbers(xml_file) for
xml_file in xml_files]
        return shapes


data_paths = [
    "ENTER_LIST_OF_PATHS",
]

datapipe = SlidingWindowDataPipe(
    image_paths=data_paths,
    window_size=250,
    step_size=250
)
dataloader = DataLoader(datapipe, batch_size=12, shuffle=True)
dataloader_iter = iter(dataloader)
```

Example Usage of dataset

```python
dataloader_iter.__next__()
```

ABS

```python
def calculate_abs(f_n, r_n):
    return np.sum(np.abs(f_n - r_n))

abs_values = {}
abs_noise_25 = {}
abs_noise_5 = {}

for key1, key2, key3 in zip(reflectance_vectors.keys(),
                            reflectance_vectors_noise_25.keys(),
                            reflectance_vectors_noise_5.keys()):
    ref_vec = denoised_reflectance_image[7:115, 1000, 200]

    abs_values[key1] = calculate_abs(ref_vec,
reflectance_vectors[key1])
    abs_noise_25[key2] = calculate_abs(ref_vec,
reflectance_vectors_noise_25[key2])
    abs_noise_5[key3] = calculate_abs(ref_vec,
reflectance_vectors_noise_5[key3])

abs_values
```

```
{'c2mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
3.0309330292966967,
 'bir1lm046_unbrecciated_lunar_meteorite__coarse_texture':
12.492682048995588,
 'c1lm36_Igneous__Lunar_Unbrecciated_Basalt_': 3.6990471664203888,
 'cclm35_Breccia__Lunar_Feldspathic_Breccia': 2.726251986159569,
 'c10l01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
8.174762611219567,
 'c3mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
4.587674998395645,
 'colm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
8.405454563394146,
 'c12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
3.8087252690275637,
 'c6mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
3.5941383237977513,
 'bir1lm048_Mare_basalt': 8.061934945022614,

'cama01_Anomalous_(ACANOM)__Antarctic_meteorite_of_presumed_lunar_orig
in': 2.998137063170352,
 'calm34_Breccia__Lunar_Feldspathic_Breccia': 31.95410963954807,
 'cflm34_Breccia__Lunar_Feldspathic_Breccia': 18.137495132223997,
 'cpmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
3.986389991094251,
 'c1lm50b_0.148_g': 17.575408278183684,
 'cclm34_Breccia__Lunar_Feldspathic_Breccia': 2.1815875116385026,
 'cblm34_Breccia__Lunar_Feldspathic_Breccia': 2.897205682939041,
 'bir1lm053_Breccia__Lunar_Feldspathic_Breccia': 4.518390249579475,
 'celm13_(E)_White_end.': 26.184883510858967,
 'bir1lm044_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
16.34594287447547,
 'c1lm52b_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
2.167923484028919,
 'c1lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
43.70341453668339,
 'c1lm51_Lunar_meteorite__feldspathic_fragmental_breccia':
9.664285571896851,
 's12lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
3.8337587275096374,
 'c1lm52c_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
13.919895536996412,
 'cclm13_Top_whitish.': 12.552244594445531,
 'cblm35_Breccia__Lunar_Feldspathic_Breccia': 3.2627331822973185,
 'c1lm50c_0.148_g': 35.33550602630124,
 'sblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
2.436514356599916,
```

 'c2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
2.9259272937509353,
 's2lm06_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
2.890376715720617,
 'bir1lm047_Mare_basalt': 3.9610863632306046,
 'sslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
8.237970697762178,
 'c1lm50_0.148_g': 22.41122467039846,
 'c1lm51c_Lunar_meteorite__feldspathic_fragmental_breccia':
43.61221132308978,
 'bir1lm054_Breccia__Lunar_Feldspathic_Breccia': 6.915736203272035,
 's10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
9.417175214001064,
 'c8mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
4.237128171488529,
 'c9mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
12.573381786384088,
 'c1lm52_Breccia__Lunar_Feldspathic_Regolith_Breccia__0.97_g':
2.272715187502431,
 'cdlm34_Breccia__Lunar_Feldspathic_Breccia': 26.979724515424618,
 'crlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
6.006470341901541,
 'cnmb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
26.09015217236132,
 'cnlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
2.36574484374201,
 'cblm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
2.432665715708545,
 'rcalm03_Ground_portion_of_LM-LAM-002': 25.506231251966177,
 'calm02_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
2.2576679480551376,
 'calm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
1.9781938407608117,
 'calm35_Breccia__Lunar_Feldspathic_Breccia': 10.001322180505204,
 'c1lm34_Breccia__Lunar_Feldspathic_Breccia': 10.93539685225542,
 'interpolated_data': 3.167139866368069,
 'cblm03_Ground_portion_of_LM-LAM-002': 25.247666391521232,
 'c10lm1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
9.454390356871226,
 'c1lm51b_Lunar_meteorite__feldspathic_fragmental_breccia':
14.979831804113843,
 'chlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
3.7540763961692414,
 's10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
8.78406592689958,
 'bir1lm045_unbrecciated_lunar_meteorite__coarse_texture':
4.477612280084221,

```
 'calm03_Ground_portion_of_LM-LAM-002': 25.513748723545742,
 'srlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
6.010284992188235,
 'cdlm35_Breccia__Lunar_Feldspathic_Breccia': 24.8019392758351,

'camt313_Igneous__Olivine_Gabbro__Purchased_from_catchafallingstar.com
': 16.150427359526468,
 'c7mb29_Silicate_(Ino)__Pyroxene__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
4.352561237609393,
 'c2lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
26.298611737891097,
 'celm34_Breccia__Lunar_Feldspathic_Breccia': 3.9615279908671144,
 'c10m01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
8.814675242147624,
 'c2lm42_Silicate_(Tecto)__Feldspar_Plagioclase__from_lunar_basalt':
43.8512628781821,
 'calmcr_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
2.6477119300012015,
 'cblm13_Mostly_black_side_opposite_of_(A).': 4.038708973972942,
 'calmn1_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
4.47659915176231,
 'c1lm41_Silicate_(Ino)__Pyroxene__from_lunar_basalt':
25.19390811498709,
 'c1lm39_Igneous__Lunar_Unbrecciated_Basalt_': 5.013335472498116,
 'comb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
26.83789952406646,
 'calmc2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
4.013957831560372,
 'c4mb29_Igneous__Gabbroic__A-
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
6.1507077908507,
 'cslm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
8.251951933707305,
 'celm35_Breccia__Lunar_Feldspathic_Breccia': 3.3125295432107156,
 'sblm03_Ground_portion_of_LM-LAM-002': 25.22508270145452,
 'cdlm13_(D)_Bottom_black_flat_with_brown_spot.': 8.68905329555699,
 'c1lm38_Breccia__Lunar_Basaltic_Breccia': 3.1671398663680694,
 'cflm35_Breccia__Lunar_Feldspathic_Breccia': 2.828681050865722,
 'solm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
8.406352338620511,
 'calmn2_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
10.378957266332014,
 'calmca_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
3.3284909270600296,
 'c1lm37_Breccia__Lunar_Basaltic_Breccia': 2.7602745149868575,
 'shlm01_Antarctic_Meteorite_of_Presumed_Lunar_Origin':
3.7537877730500524,
 'cqmb29_Igneous__Gabbroic__A-
```

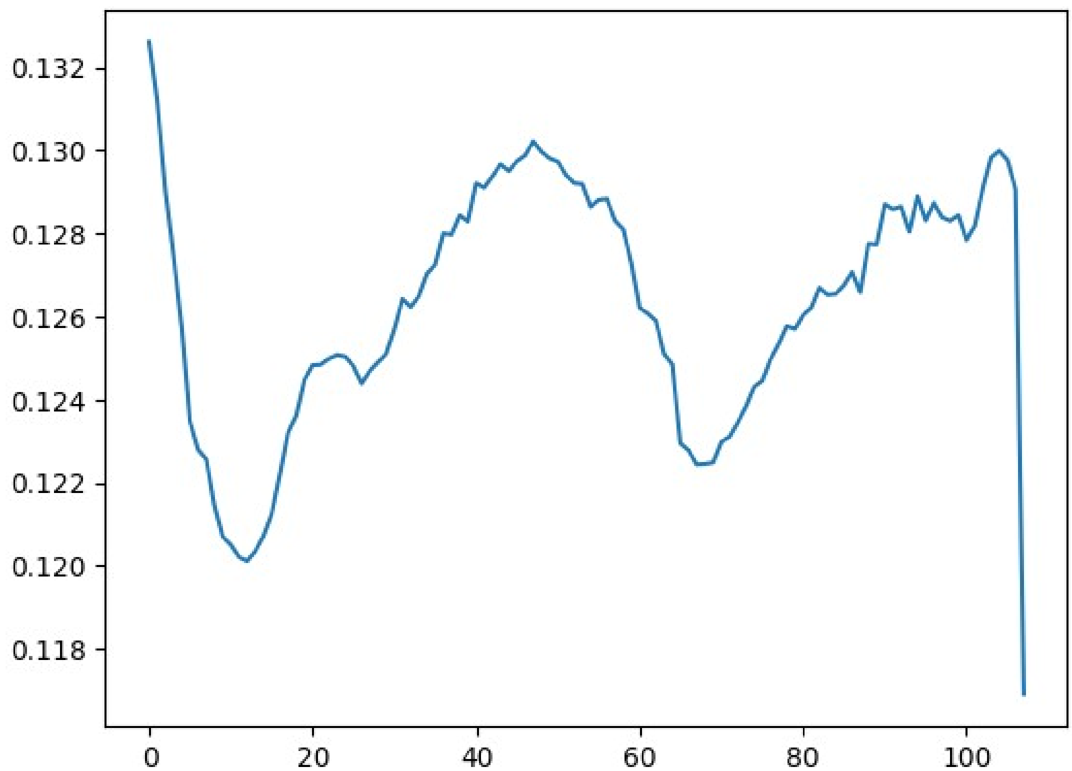
```
881757_Lunar_Gabbrocontaining_Plag__Pyroxene_and_Ilmenite':
33.100889599777425}
```

```
temp = min(abs_values.values())
res = [key for key in abs_values if abs_values[key] == temp]

temp25 = min(abs_noise_25.values())
res25 = [key for key in abs_noise_25 if abs_noise_25[key] == temp25]

temp5 = min(abs_noise_5.values())
res5 = [key for key in abs_noise_5 if abs_noise_5[key] == temp5]

print(temp, res)
print(temp25, res25)
print(temp5, res5)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call
last)
Cell In[37], line 1
----> 1 prinarrowt(temp, res)
      2 print(temp25, res25)
      3 print(temp5, res5)

NameError: name 'prinarrowt' is not defined
```

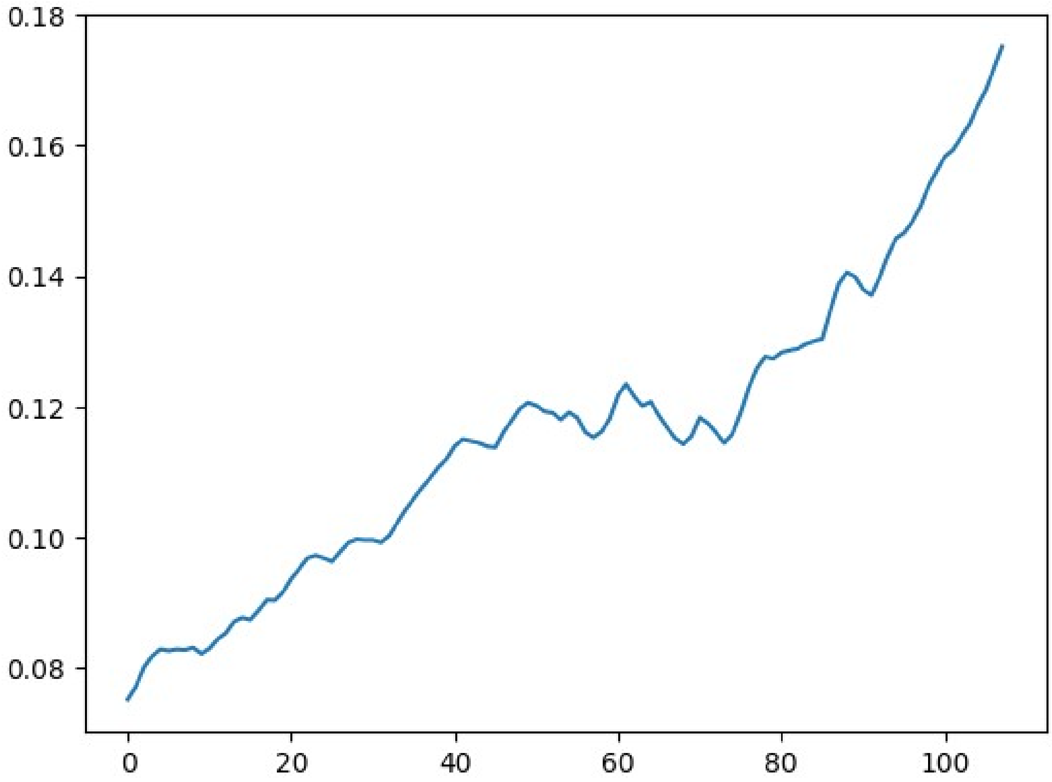
```
plt.plot(reflectance_vectors['c1lm52b_Breccia__Lunar_Feldspathic_Regol
ith_Breccia__0.97_g'])
```

```
[<matplotlib.lines.Line2D at 0x7fc82ecbd960>]
```

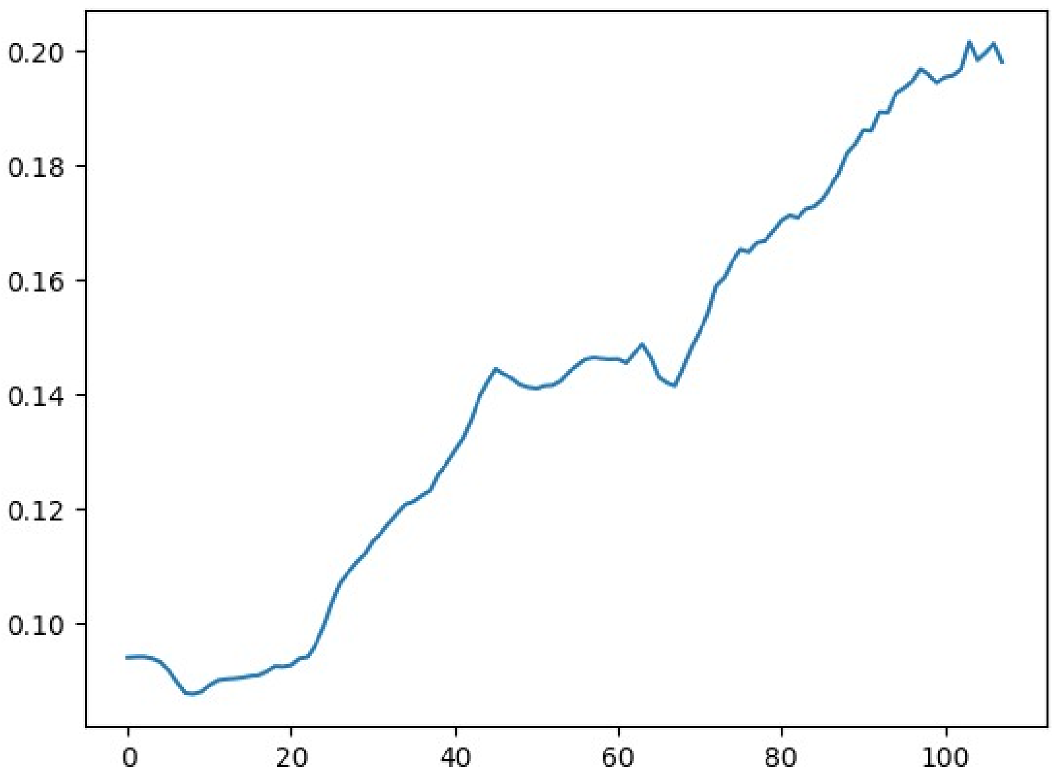



```
plt.plot(ref_vec)
```

```
[<matplotlib.lines.Line2D at 0x7fc82f8e62c0>]
```




```
diff = {}

for key in abs_noise_25.keys():
    diff[key] = abs_noise_25[key] + cprms_noise_25[key] * 100

temp25 = min(diff.values())
res25 = [key for key in diff if diff[key] == temp25]

res25

['calmcr_Antarctic_Meteorite_of_Presumed_Lunar_Origin']

plt.plot(reflectance_vectors_noise_25['calmcr_Antarctic_Meteorite_of_P
resumed_Lunar_Origin'])

[<matplotlib.lines.Line2D at 0x7fc82ed7b4f0>]
```


0.20
0.18
0.16
0.14
0.12
0.10
0
20
40
60
80
100