SFT REPORT

Sediment accumulation investigation based on sediment calculation at diverse location of Dove Elbe

DEEPAK BOGATI (6066481)

SUBMITTED TO Ellen Werner, Mona Lütjens, Annette Scheider

INTRODUCTION

Python is a programming language that is widely used in the software industry and academia. It is known for its simplicity, readability, and flexibility, which make it a popular choice for beginners and experienced programmers alike. Python has a large and active community of users, which has contributed to the development of a wide range of libraries and frameworks that support a variety of programming tasks, including data analysis, machine learning, and scientific computing. In this project, we are analyzing a Multibeam and single beam data form DV Ocean survey done on 2nd of july 2021 using python in jupiternote book.

After cleaning the noise form the data in qimera, x,y,z export from qimera is imported in jupiter notebook for volume calculation of the sediments in five diefferent target locations.

Importing Necessary Liberaries

In [21]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from shapely.geometry import Polygon
```

sample 1

Importing x,y,z Low Frequency singlebeam data form qimera for location/Line 1 and

Also removed the outliers form the height data for final vertical(z) data for low frequancy data

In [2]:

```
#LfL1 (short form for lowfrequensy SBES data for line 1)
LfL1 = pd.read_csv('LfL1.csv')
df = pd.read_csv('LfL1.csv')
df['Footprint Z'].describe()
mean = df['Footprint Z'].mean()
std = df['Footprint Z'].std()
threshold = 3 * std
outliers = df.loc[np.abs(df['Footprint Z'] - mean) > threshold]
df = df.drop(outliers.index)
df.to_csv('cleaned_data.csv', index=False)
df
```

Out[2]:

	#Date Time	Footprint X	Footprint Y	Footprint Z
0	2021-07-02 14:06:35.697	568995.323	5931523.039	7.401
1	2021-07-02 14:06:35.785	568995.393	5931523.251	7.399
2	2021-07-02 14:06:35.874	568995.476	5931523.463	7.397
3	2021-07-02 14:06:35.963	568995.541	5931523.712	7.905
4	2021-07-02 14:06:36.051	568995.625	5931523.920	7.903
...	...	...	...	...
1949	2021-07-02 14:09:29.211	569535.836	5931626.613	4.948
1950	2021-07-02 14:09:29.300	569536.122	5931626.647	4.940
1951	2021-07-02 14:09:29.388	569536.405	5931626.689	4.974
1952	2021-07-02 14:09:29.477	569536.692	5931626.753	4.962
1953	2021-07-02 14:09:29.565	569536.975	5931626.798	4.954

1954 rows × 4 columns

Importing x,y,z High frequency singlebeam data form qimera for location/Line 1 and

Also removed the outliers form the height data for final vertical(z) data for High frequamcy data

In [22]:

```
#HlF1 (High frequancy sbes data for line 1)
HfL1 = pd.read_csv('HfL1.csv')
df2 = pd.read_csv('HfL1.csv')
df2['Footprint Z'].describe()
mean = df['Footprint Z'].mean()
std = df['Footprint Z'].std()
threshold = 3 * std
outliers = df2.loc[np.abs(df2['Footprint Z'] - mean) > threshold]
df2 = df2.drop(outliers.index)
df2.to_csv('cleaned_data.csv', index=False)
df2
```

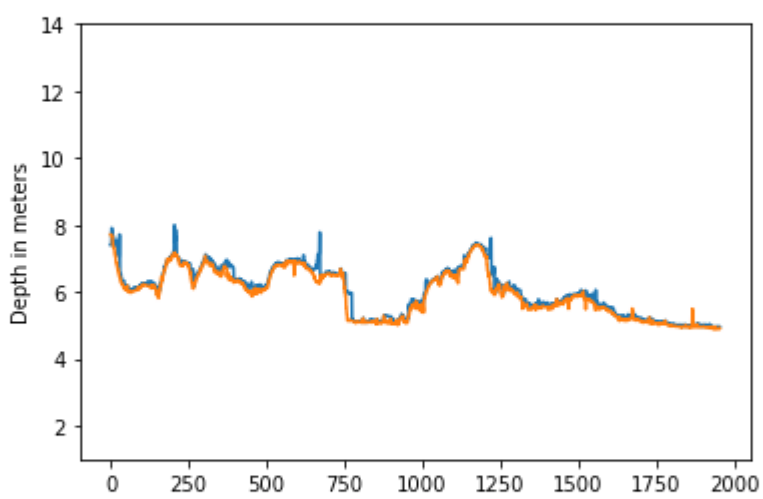
Out[22]:

	#Date Time	Footprint X	Footprint Y	Footprint Z
0	2021-07-02 14:06:35.698	568995.238	5931523.007	7.717

	#Date Time	Footprint X	Footprint Y	Footprint Z
1	2021-07-02 14:06:35.786	568995.307	5931523.219	7.714
2	2021-07-02 14:06:35.875	568995.390	5931523.441	7.662
3	2021-07-02 14:06:35.963	568995.474	5931523.650	7.661
4	2021-07-02 14:06:36.051	568995.558	5931523.858	7.678
...	...	...	...	...
1949	2021-07-02 14:09:29.211	569535.731	5931626.592	4.903
1950	2021-07-02 14:09:29.300	569536.017	5931626.636	4.895
1951	2021-07-02 14:09:29.388	569536.300	5931626.669	4.909
1952	2021-07-02 14:09:29.477	569536.587	5931626.733	4.898
1953	2021-07-02 14:09:29.565	569536.870	5931626.778	4.909

In [23]:

```
# Loading the data into a data frame
df = pd.read_csv('LfL1.csv')
df2 = pd.read_csv('HfL1.csv')

# Extracting the columns to be plotted
low = df['Footprint Z']
high = df2['Footprint Z']
low.mean()

plt.plot(low)
plt.plot(high)
plt.ylim(1, 14)

# Adding a label to the y-axis
plt.ylabel('Depth in meters')

# Show the plot
plt.show()
thickness = df["Footprint Z"] - df2["Footprint Z"]
thickness = thickness.mean()
thickness
```



0.11097031729785035

In [5]:
```
a = low.mean()
b = high.mean()
a-b
```

Out[5]: 0.11097031729785822

The approximate bulk density of wet clay that is commonly used in normal-weight concrete is between 1240-1410 kg/m3. considering 1300kg/m3 density for the sample sediment

In [6]:
```
#area of the target survey areas imported from Qimera.
area1 = 8050.50
area2 = 5443.01
area3 = 1423.81
area4 = 393.31
area5 = 12155.63
```

In [24]:
```
#Calculation of sediment mass per unit area.
density = 1300
volume = area1 * thickness
mass = density*volume
mass
sediment_mass_per_unit_area = mass/area1
sediment_mass_per_unit_area
```

Out[24]: 144.26141248720546

Sample 2

Importing x,y,z Low Frequency singlebeam data form qimera for location/Line 2 and

Also removed the outliers form the height data for final vertical(z) data for low frequancy data

In [26]:
```
df3 = pd.read_csv('LfL2.csv')
df4 = pd.read_csv('HfL2.csv')
df3
```

Out[26]:

	#Date Time	Footprint X	Footprint Y	Footprint Z
0	2021-07-02 13:55:45.165	570296.771	5929320.827	7.530
1	2021-07-02 13:55:45.255	570296.774	5929321.205	7.530
2	2021-07-02 13:55:45.342	570296.774	5929321.570	7.507
3	2021-07-02 13:55:45.430	570296.784	5929321.938	7.510
4	2021-07-02 13:55:45.519	570296.799	5929322.321	7.479
...	...	...	...	...

	#Date Time	Footprint X	Footprint Y	Footprint Z
1696	2021-07-02 13:58:16.786	570307.263	5929973.345	6.878
1697	2021-07-02 13:58:16.874	570307.191	5929973.717	6.876
1698	2021-07-02 13:58:16.964	570307.118	5929974.086	7.128
1699	2021-07-02 13:58:17.052	570307.035	5929974.458	7.130
1700	2021-07-02 13:58:17.138	570306.975	5929974.824	7.247

In [25]:

```
df4
```

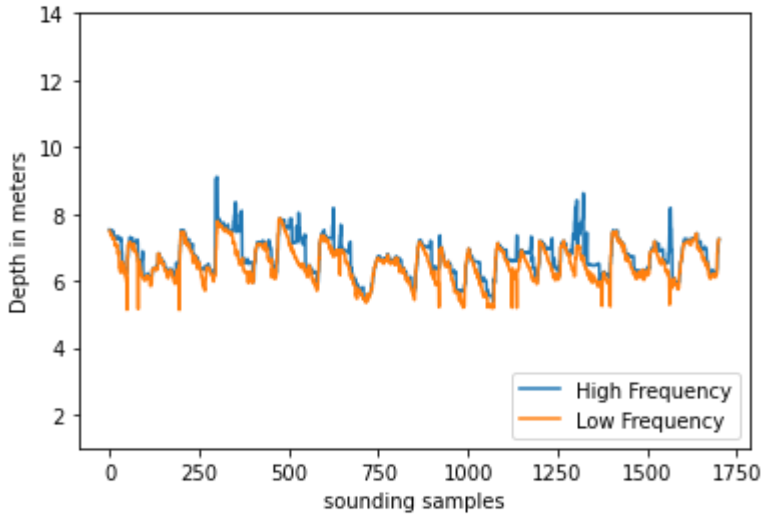
Out[25]:

	#Date Time	Footprint X	Footprint Y	Footprint Z
0	2021-07-02 12:44:34.887	573018.903	5926630.843	5.798
1	2021-07-02 12:44:35.066	573019.333	5926630.766	5.737
2	2021-07-02 12:44:35.246	573019.755	5926630.702	5.745
3	2021-07-02 12:44:35.427	573020.180	5926630.626	5.724
4	2021-07-02 12:44:35.607	573020.602	5926630.549	5.623
...	...	...	...	...
1789	2021-07-02 12:47:18.541	573397.268	5926536.073	9.283
1790	2021-07-02 12:47:18.631	573397.474	5926536.021	9.372
1791	2021-07-02 12:47:18.720	573397.664	5926535.943	9.376
1792	2021-07-02 12:47:18.811	573397.874	5926535.892	9.443
1793	2021-07-02 12:47:18.900	573398.077	5926535.818	9.444

1794 rows × 4 columns

In [27]:

```
# Extracting the columns to be plotted
low = df3['Footprint Z']
high = df4['Footprint Z']
# Adding a label to the y-axis
plt.ylim(1, 14)

plt.ylabel('Depth in meters')
plt.xlabel('sounding samples')

plt.plot(low)
plt.plot(high)

plt.legend(["High Frequency", "Low Frequency"], loc ="lower right")

# Show the plot
plt.show()
thickness = df3["Footprint Z"] - df4["Footprint Z"]
thickness = thickness.mean()
thickness
```



Out[27]: 0.23280834803056988

In [11]:
```
#Calculation of sediment mass per unit area.
density = 1300
volume = area2 * thickness
mass = density*volume
sediment_mass_per_unit_area = mass/area2
sediment_mass_per_unit_area
```

Out[11]: 302.65085243974085

In [28]:
```
a = low.mean()
b = high.mean()
#a
```

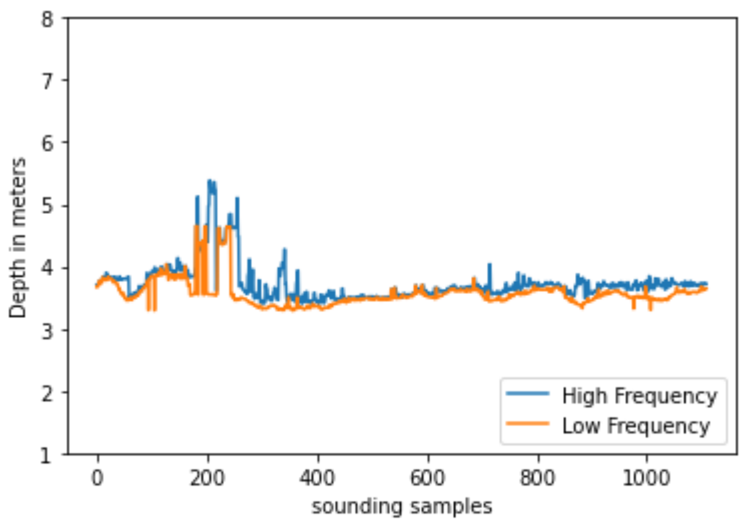
Sample 3

Importing x,y,z Low Frequency singlebeam data form qimera for location/Line 3 and

Also removed the outliers form the height data for final vertical(z) data for low frequancy data

In [31]:

```
df3 = pd.read_csv('LfL3.csv')
df4 = pd.read_csv('HfL3.csv')
df3
df4

# Extracting the columns to be plotted
low = df3['Footprint Z']
high = df4['Footprint Z']
# Adding a label to the y-axis
plt.ylim(1, 8)

plt.ylabel('Depth in meters')
plt.xlabel('sounding samples')

plt.plot(low)
plt.plot(high)

plt.legend(["High Frequency", "Low Frequency"], loc ="lower right")

# Show the plot
plt.show()
thickness = df3["Footprint Z"] - df4["Footprint Z"]
thickness = thickness.mean()
thickness
```



Out[31]: 0.15812533814247046

In [29]:

```
a = low.mean()
b = high.mean()
#b
```

In [32]:

```
#Calculation of sediment mass per unit area.
density = 1300
volume = area3 * thickness
mass = density*volume
sediment_mass_per_unit_area = mass/area3
sediment_mass_per_unit_area
```

Out[32]: 205.5629395852116

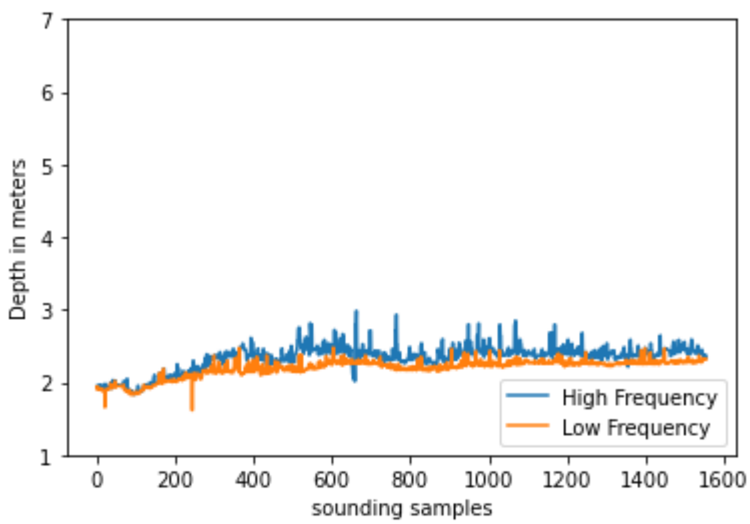
Sample 4

Importing x,y,z Low Frequency singlebeam data form qimera for location/Line 4 and

Also removed the outliers form the height data for final vertical(z) data for low frequancy data

In [33]:

```
df3 = pd.read_csv('LfL4.csv')
df4 = pd.read_csv('HfL4.csv')
#df3
#df4

# Extracting the columns to be plotted
low = df3['Footprint Z']
high = df4['Footprint Z']
# Adding a label to the y-axis
plt.ylim(1, 7)

plt.ylabel('Depth in meters')
plt.xlabel('sounding samples')

plt.plot(low)
plt.plot(high)

plt.legend(["High Frequency", "Low Frequency"], loc ="lower right")

# Show the plot
plt.show()
thickness = df3["Footprint Z"] - df4["Footprint Z"]
thickness = thickness.mean()
thickness
```



Out[33]: 0.1426304627249356

In [34]:

```
#Calculation of sediment mass per unit area.
density = 1300
volume = area4 * thickness
mass = density*volume
sediment_mass_per_unit_area = mass/area4
sediment_mass_per_unit_area
```

Out[34]:

```
185.41960154241627
```

In [18]:

```
a = low.mean()
b = high.mean()
#a
```

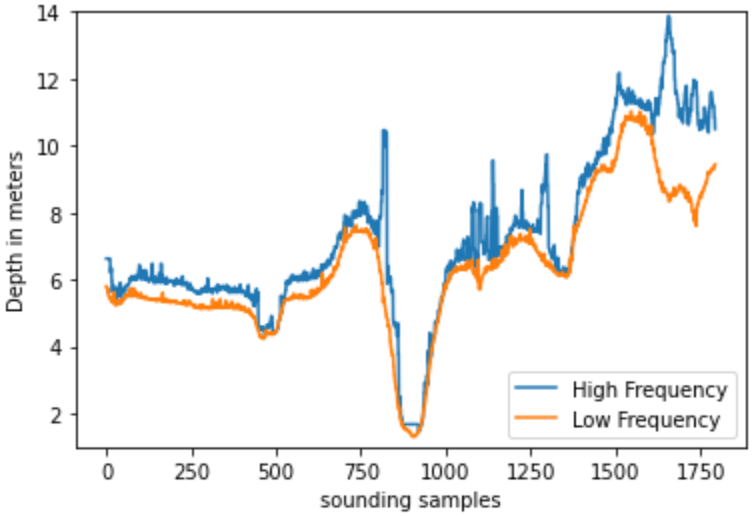
Sample 5

Importing x,y,z Low and high Frequency singlebeam data form qimera for location/Line 5 and

Also removed the outliers form the height data for final vertical(z) data for low frequancy data

In [35]:

```
df3 = pd.read_csv('LfL5.csv')
df4 = pd.read_csv('HfL5.csv')
df3
df4

# Extracting the columns to be plotted
low = df3['Footprint Z']
high = df4['Footprint Z']
# Adding a label to the y-axis
plt.ylim(1, 14)

plt.ylabel('Depth in meters')
plt.xlabel('sounding samples')

plt.plot(low)
plt.plot(high)

plt.legend(["High Frequency", "Low Frequency"], loc ="lower right")

# Show the plot
plt.show()
thickness = df3["Footprint Z"] - df4["Footprint Z"]
thickness = thickness.mean()
thickness
```



Out[35]: 0.8340356943669817

In [36]:

```
#Calculation of sediment mass per unit area.
density = 1300
volume = area5 * thickness
mass = density*volume
sediment_mass_per_unit_area = mass/area5
sediment_mass_per_unit_area
```

Out[36]: 1084.2464026770763

In [ ]:

In [ ]: