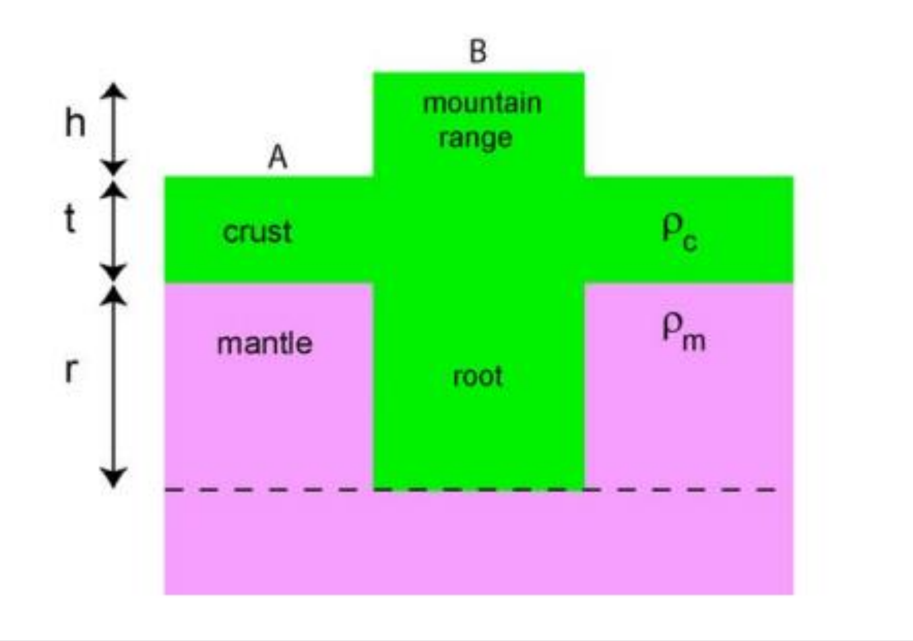
DETERMINING ISOSTATIC COMPENSATION

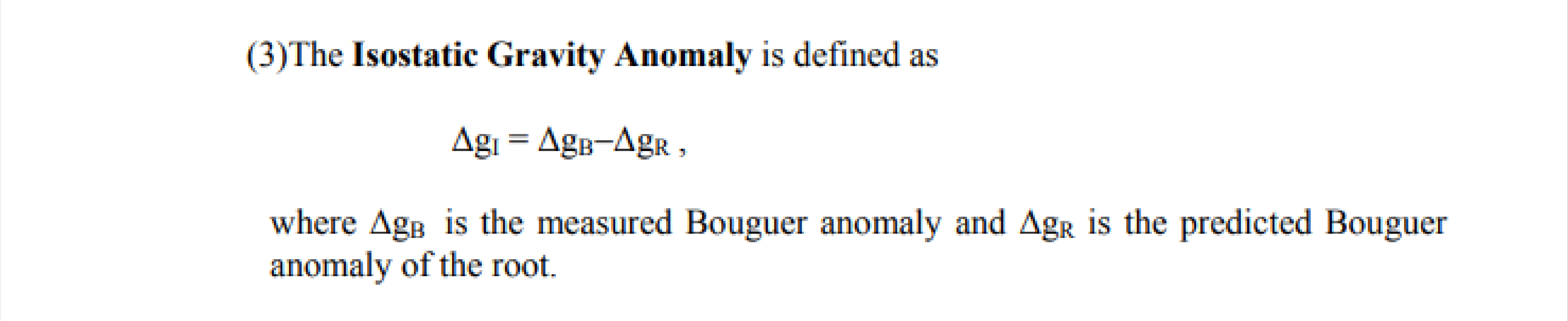
Rohit Chaurasia (23n0011)
Mahendra Chaudhari (23n0004)
Ashish Singh (23n0010)

Airy -Heiskanen model of Isostasy :



Different topographic heights are accommodated by changes in crustal thickness, in which the crust has a constant density.

Isostatic Gravity Anomaly :



(3)The **Isostatic Gravity Anomaly** is defined as

$$\Delta g_I = \Delta g_B - \Delta g_R \ ,$$

where Δg_B is the measured Bouguer anomaly and Δg_R is the predicted Bouguer anomaly of the root.

Determining Compensation :

Overcompensation $\Delta g < 0$

Under compensation $\Delta g > 0$

Complete compensation $\Delta g = 0$

What we have done

01

Fetch the data

02

Apply isostatic anomaly and determine compensation.

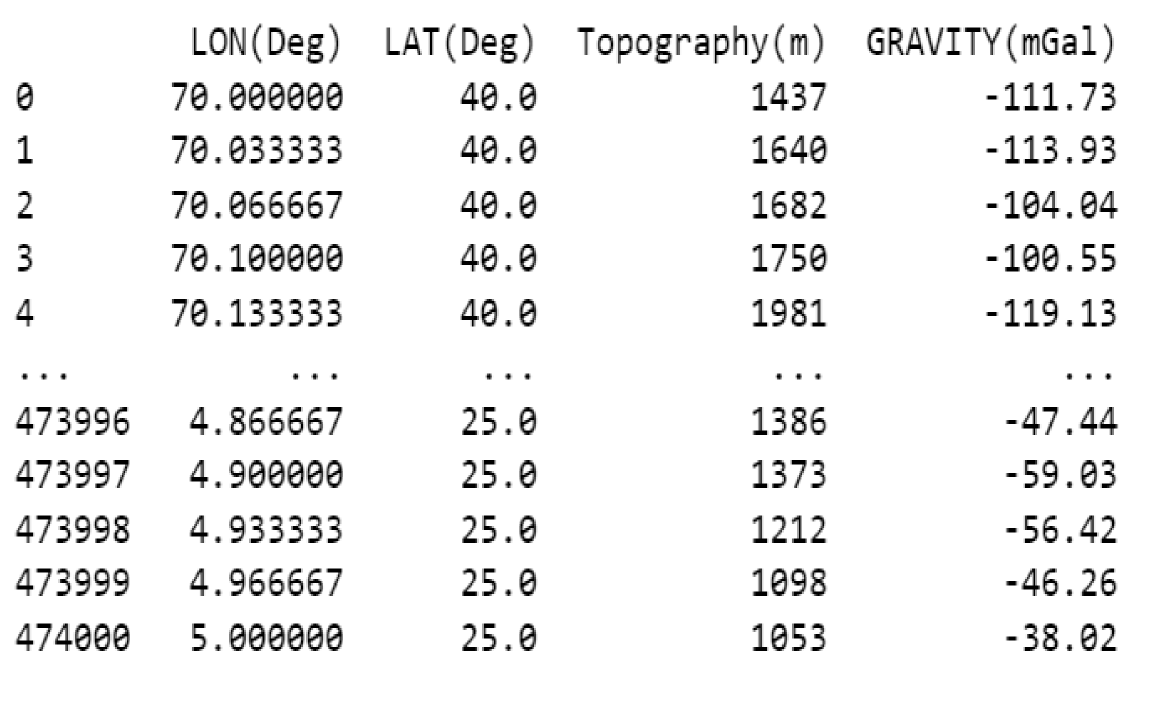
03

Plot the isostatic compensation on map

Fetching the data :

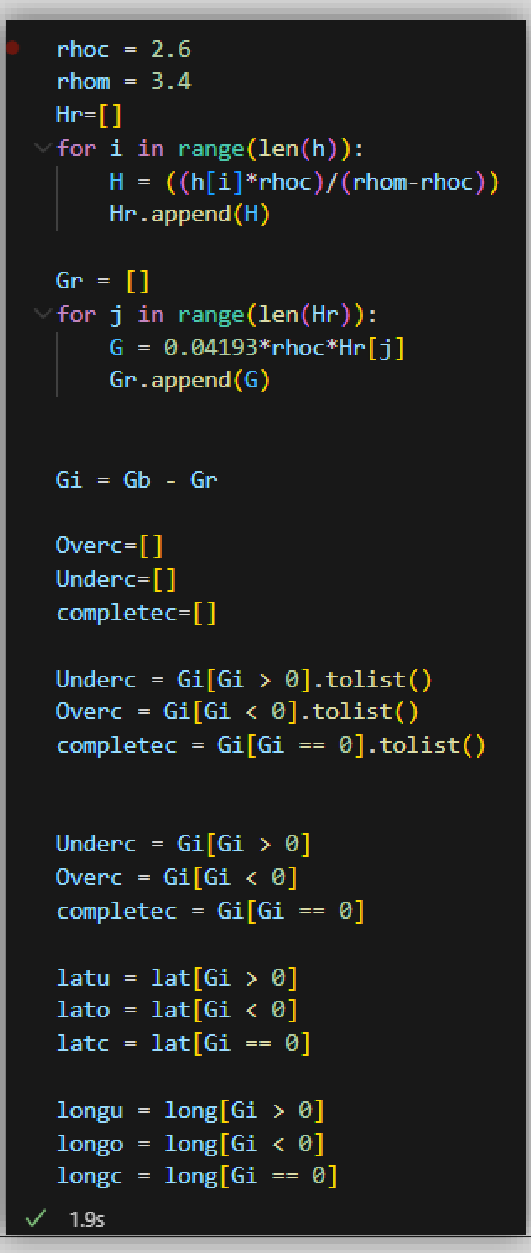
Range of data :

Longitude: 99.96 – 70 AND 0 - 5
Latitude : 40 - 25



```
         LON(Deg)  LAT(Deg)  Topography(m)  GRAVITY(mGal)
0       70.000000      40.0           1437        -111.73
1       70.033333      40.0           1640        -113.93
2       70.066667      40.0           1682        -104.04
3       70.100000      40.0           1750        -100.55
4       70.133333      40.0           1981        -119.13
...           ...       ...            ...            ...
473996   4.866667      25.0           1386         -47.44
473997   4.900000      25.0           1373         -59.03
473998   4.933333      25.0           1212         -56.42
473999   4.966667      25.0           1098         -46.26
474000   5.000000      25.0           1053         -38.02
```

Behind the scene :



```
rhoc = 2.6
rhom = 3.4
Hr=[]
for i in range(len(h)):
    H = ((h[i]*rhoc)/(rhom-rhoc))
    Hr.append(H)

Gr = []
for j in range(len(Hr)):
    G = 0.04193*rhoc*Hr[j]
    Gr.append(G)


Gi = Gb - Gr

Overc=[]
Underc=[]
completec=[]

Underc = Gi[Gi > 0].tolist()
Overc = Gi[Gi < 0].tolist()
completec = Gi[Gi == 0].tolist()


Underc = Gi[Gi > 0]
Overc = Gi[Gi < 0]
completec = Gi[Gi == 0]

latu = lat[Gi > 0]
lato = lat[Gi < 0]
latc = lat[Gi == 0]

longu = long[Gi > 0]
longo = long[Gi < 0]
longc = long[Gi == 0]
```

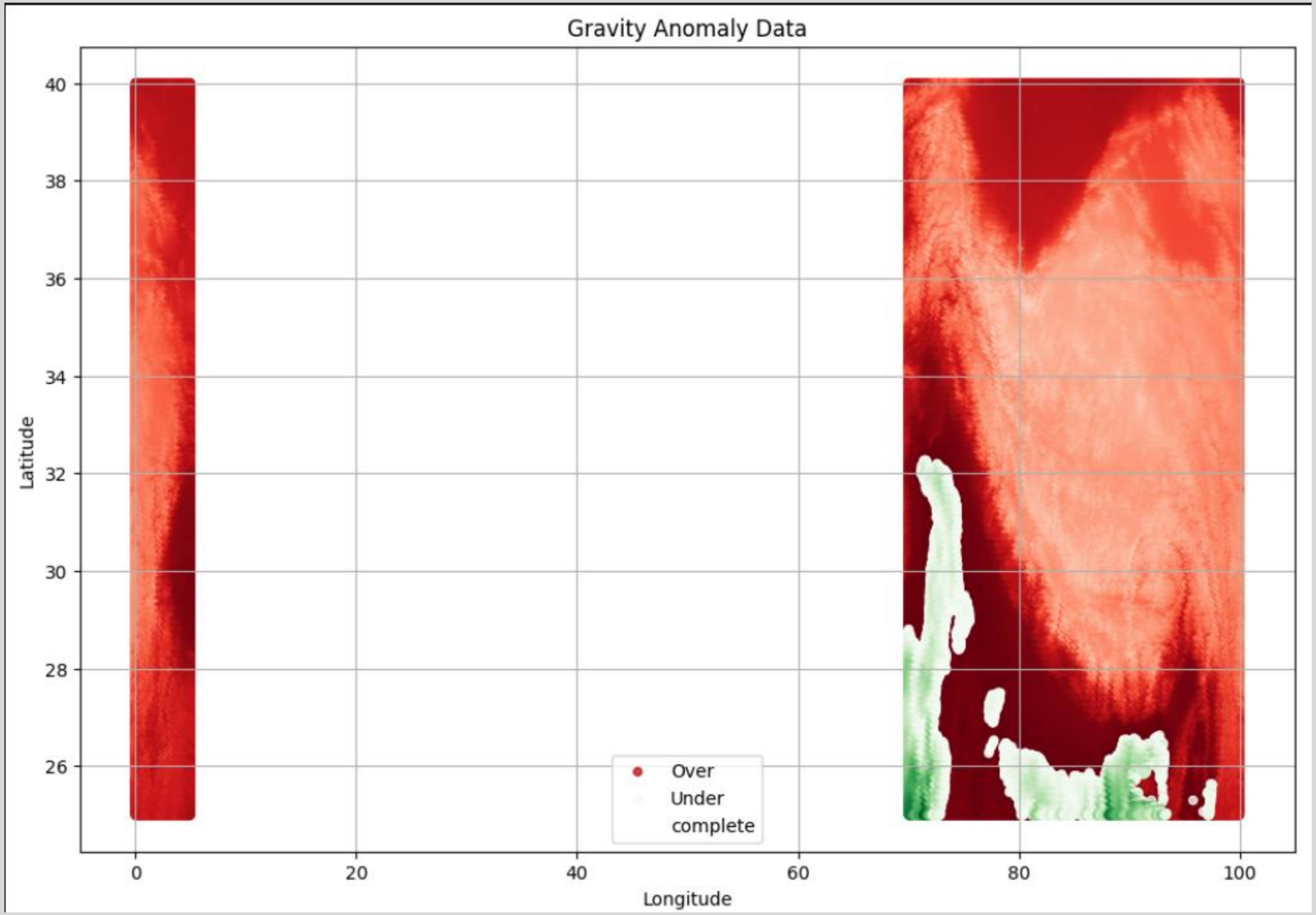
✓ 1.9s

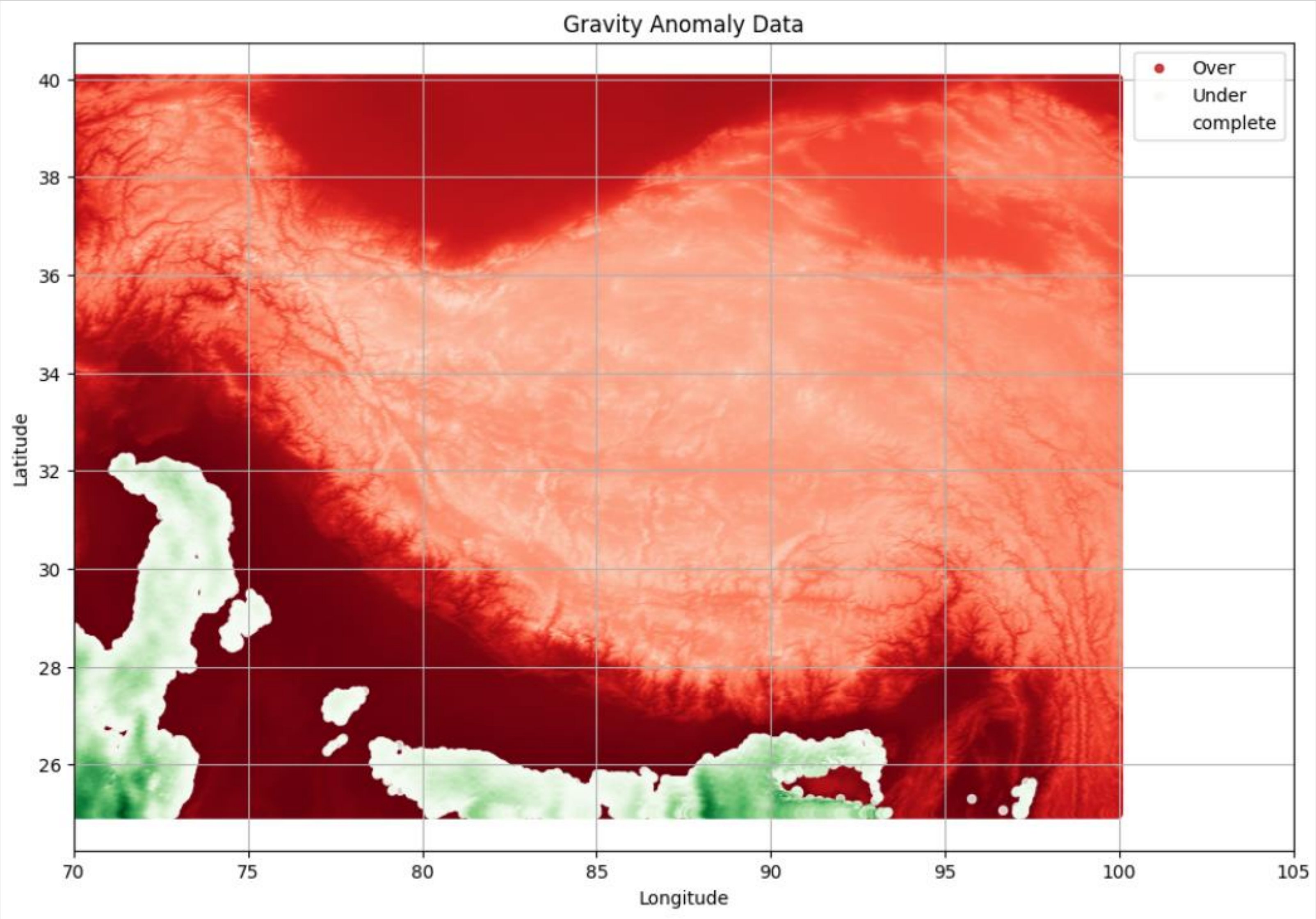
To find roots

To find the root Bouguer Anomaly

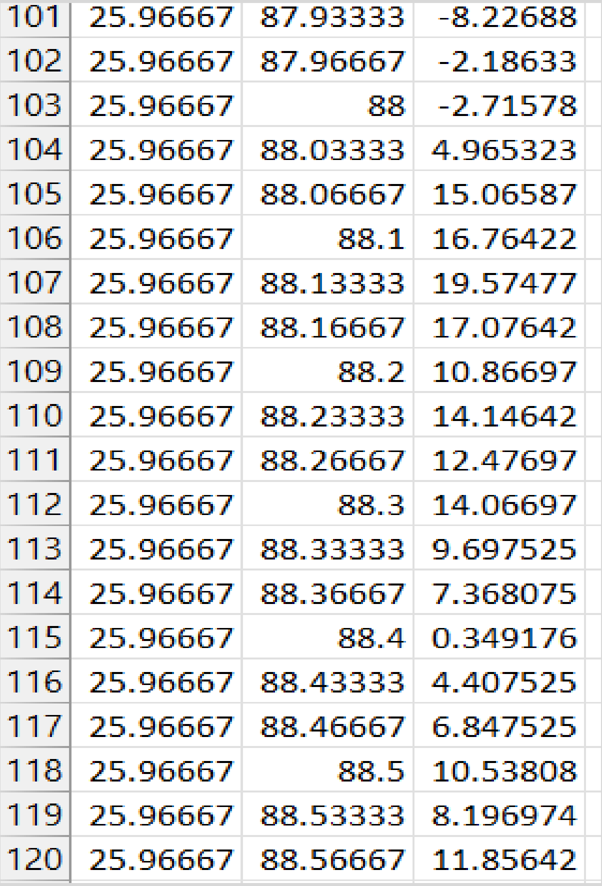
To determine Compensation

What we get :


Gravity Anomaly Data
Latitude
40
38
36
34
32
30
28
26
Longitude
0
20
40
60
80
100
● Over
● Under
complete

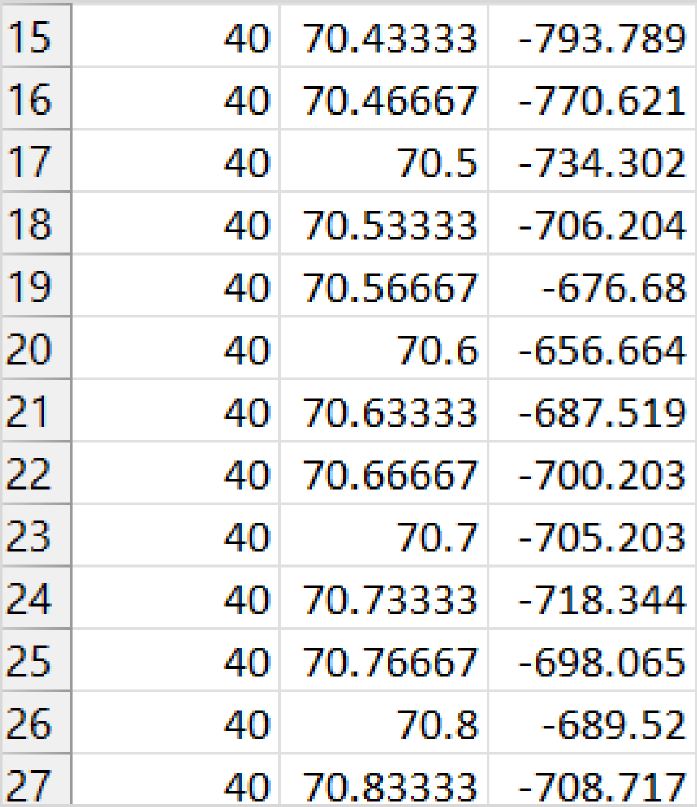

Gravity Anomaly Data
● Over
● Under
complete
Latitude
40
38
36
34
32
30
28
26
Longitude
70
75
80
85
90
95
100
105

Under compensation



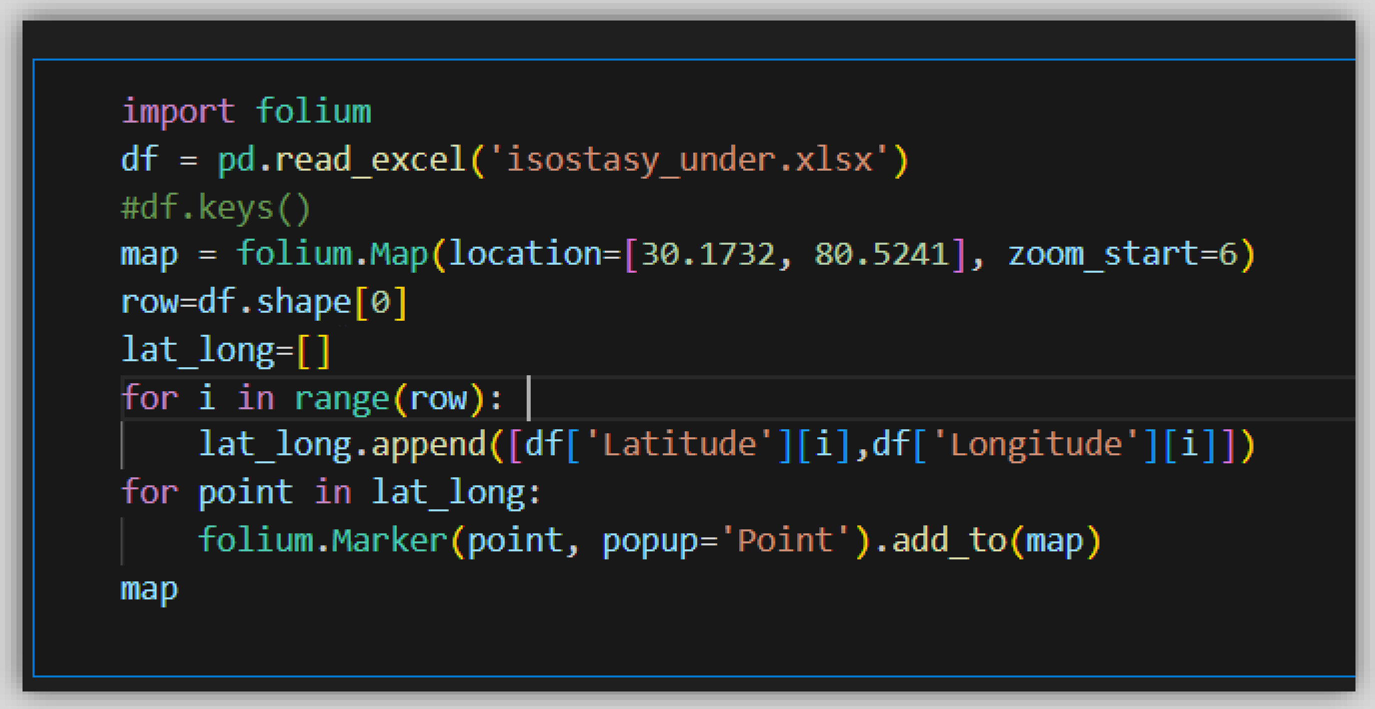
101	25.96667	87.93333	-8.22688
102	25.96667	87.96667	-2.18633
103	25.96667	88	-2.71578
104	25.96667	88.03333	4.965323
105	25.96667	88.06667	15.06587
106	25.96667	88.1	16.76422
107	25.96667	88.13333	19.57477
108	25.96667	88.16667	17.07642
109	25.96667	88.2	10.86697
110	25.96667	88.23333	14.14642
111	25.96667	88.26667	12.47697
112	25.96667	88.3	14.06697
113	25.96667	88.33333	9.697525
114	25.96667	88.36667	7.368075
115	25.96667	88.4	0.349176
116	25.96667	88.43333	4.407525
117	25.96667	88.46667	6.847525
118	25.96667	88.5	10.53808
119	25.96667	88.53333	8.196974
120	25.96667	88.56667	11.85642

Over compensation



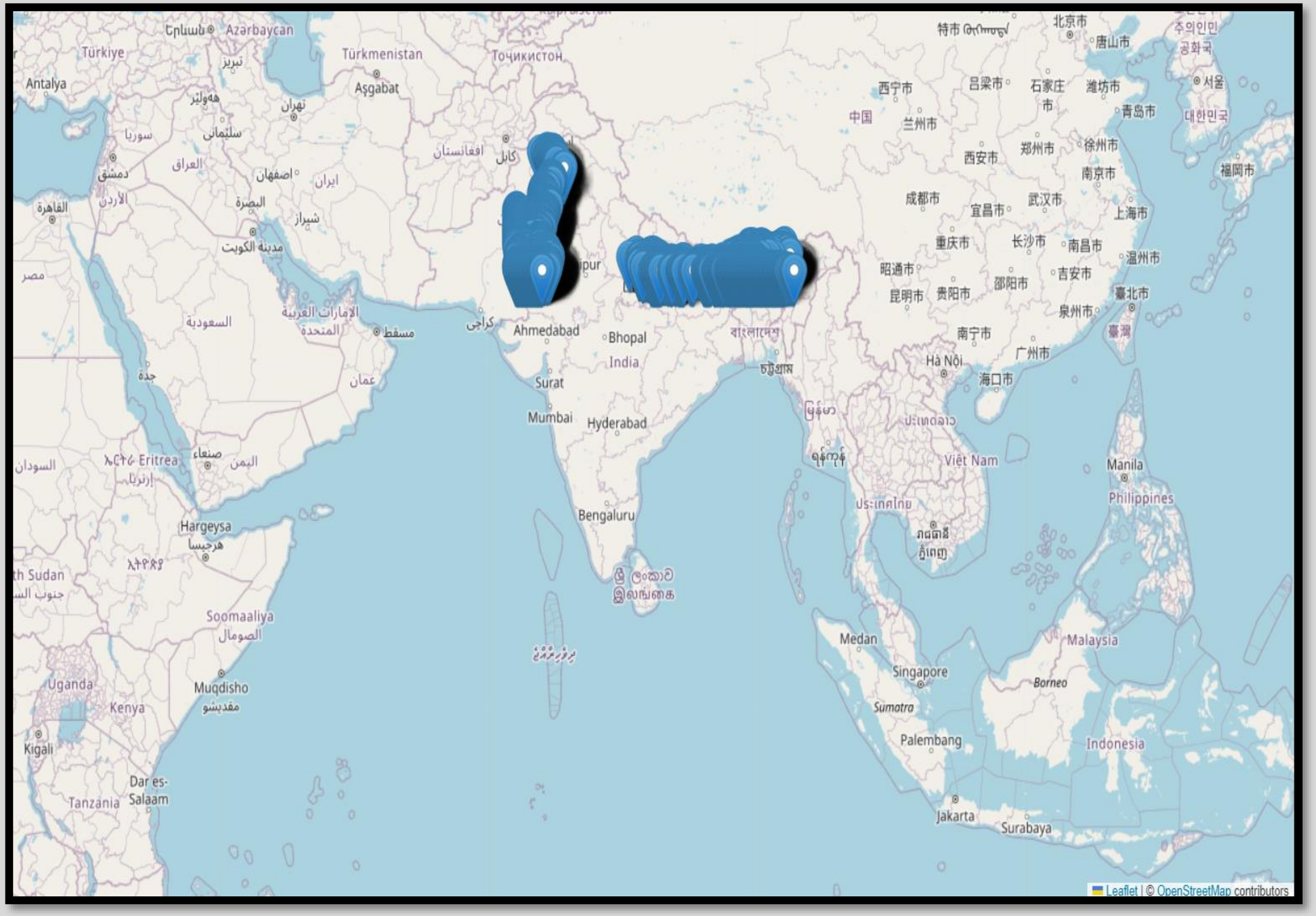
15	40	70.43333	-793.789
16	40	70.46667	-770.621
17	40	70.5	-734.302
18	40	70.53333	-706.204
19	40	70.56667	-676.68
20	40	70.6	-656.664
21	40	70.63333	-687.519
22	40	70.66667	-700.203
23	40	70.7	-705.203
24	40	70.73333	-718.344
25	40	70.76667	-698.065
26	40	70.8	-689.52
27	40	70.83333	-708.717

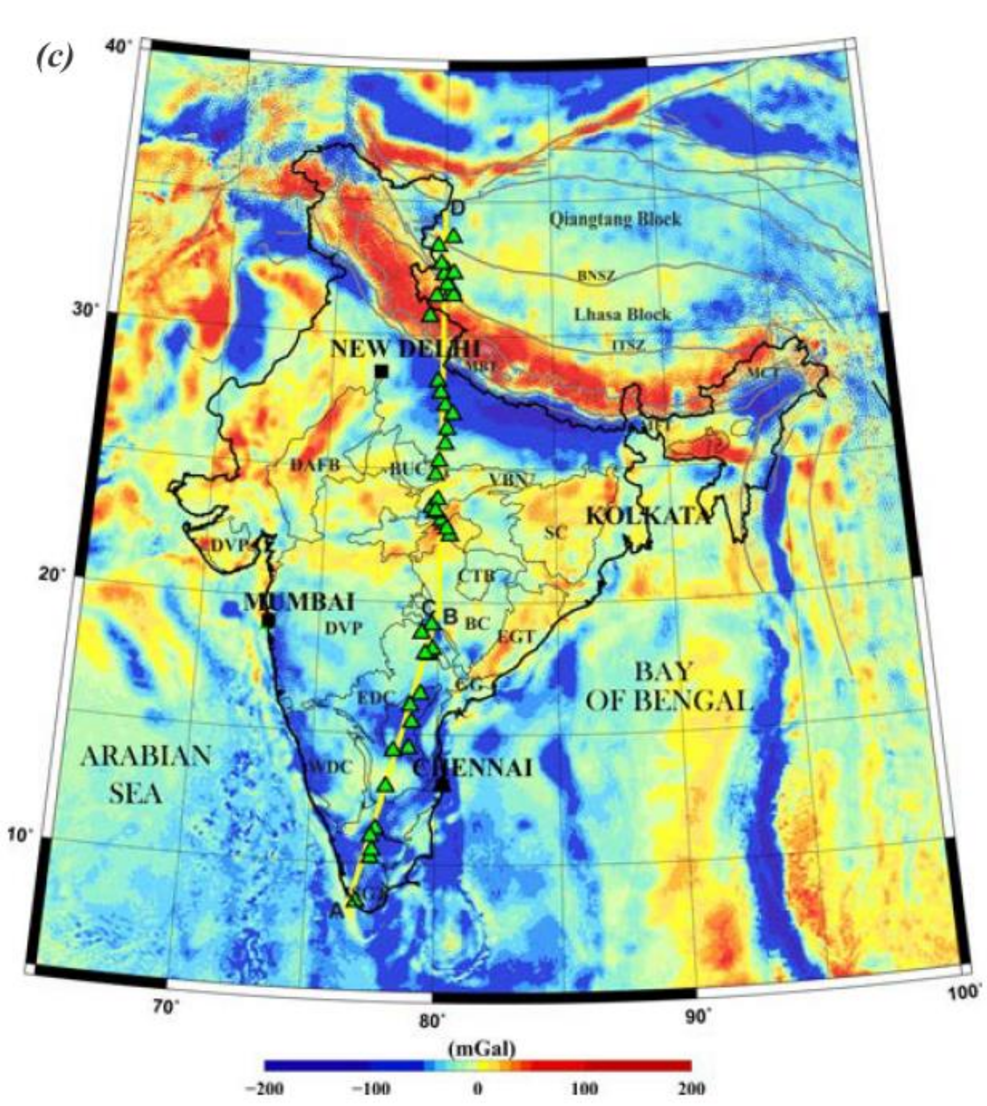
To map under Compensation :



```
import folium
df = pd.read_excel('isostasy_under.xlsx')
#df.keys()
map = folium.Map(location=[30.1732, 80.5241], zoom_start=6)
row=df.shape[0]
lat_long=[]
for i in range(row):
    lat_long.append([df['Latitude'][i],df['Longitude'][i]])
for point in lat_long:
    folium.Marker(point, popup='Point').add_to(map)
map
```

On Map: Over Compensation





Validating from real data

Reference : Niraj Kumar, A.P. Singh, V.M. Tiwari* Gravity Anomalies, Isostasy and Density Structure of the Indian Continental Lithosphere