

Step 3: Perform Calculation of Basic Statistics based on Lake Michigan Imagery Data

After reconstructing the Lake Michigan area data into 2-Dimension, we will attempt to calculate the mean, standard deviation, skewness, centroid, and kurtosis along the directions of latitude and longitude.

```
In [1]: # cited source: https://stackoverflow.com/questions/13894631/image-skewness-
```

```
In [2]: import os
import pandas as pd
import scipy
import numpy as np
from tqdm import tqdm
```

```
In [3]: os.getcwd()
os.chdir("/srv/scratch/NOAA/GOES_Hourly_Statistics/zone_0_T_2013Fall_2014Spr
os.getcwd()
```

```
Out[3]: '/srv/scratch/NOAA/GOES_Hourly_Statistics/zone_0_T_2013Fall_2014Spring'
```

```
In [4]: len(os.listdir())
```

```
Out[4]: 15607
```

```
In [5]: filename = os.listdir()
filename.sort()
len(filename)
```

```
Out[5]: 15607
```

```
In [6]: # for fn in filename:
#     time_f = str(fn[20:22])
#     print(time_f)
# list_da.append(date_f)
```

```
In [7]: dataset = pd.read_csv("T_goes15.2013.12.18.1700.v01.nc-var1-t0.csv.csv", inc
dataset
```

```
Out[7]:
```

	-87.9	-87.86	-87.82	-87.78	-87.74	-87.7	-87.66	-87.62	-87.58	-87.54	...	-87.5
41.78	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.5725	...	0.5725
41.82	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.5725	...	0.5725
41.86	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.9075	0.7650	...	0.7650
41.90	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.7575	0.7750	...	0.7750
41.94	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.61	0.6450	0.6200	...	0.6200
...
45.82	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0.3
45.86	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0.3
45.90	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0.3
45.94	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0.3
45.98	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0.3

106 rows × 79 columns

```
In [8]: # df_reference_1D = pd.read_csv('02-05-2023/zone_0_sample_take_2/' + 'goes15
# df_reference_1D.head(5)
```

```
In [9]: # table_1D_len = df_reference_1D.shape[0]
# print(table_1D_len)
```

```
In [10]: # df_lake_1D_map = df_reference_1D[['latitude', 'longitude']].copy()
# df_lake_1D_map.head(5)
```

```
In [11]: # lat_list = df_lake_1D_map['latitude'].tolist()
# lon_list = df_lake_1D_map['longitude'].tolist()
```

```
In [12]: # output_dir = 'output/'
# output_csv_name = 'lat_long_1D_labels_for_plotting.csv'

# output_file_path = os.path.join(output_dir, output_csv_name)

# df_lake_1D_map.to_csv(output_file_path, index=False)
```

```
In [13]: def lake_1D_matcher(df_temp, df_lake_1D_map):
# left join df_temp with df_lake_1D_map based on latitude and longitude
df_merged_temp = pd.merge(df_lake_1D_map, df_temp, on=['latitude', 'longitude'])

# extract value column into a list
value_temp = df_merged_temp['value'].tolist()

return value_temp
```

```
In [ ]:
```

```
In [14]: lon_v = [float(x) for x in dataset.columns.to_list()]
print(len(lon_v), max(lon_v), min(lon_v))
```

```
79 -84.78 -87.9
```

```
In [15]: #for i in range(int(min(lon_v)*100), int((max(lon_v)+0.04)*100), 4):
#         print(i/100)
```

```
In [16]: lat_v = [float(x) for x in dataset.index.to_list()]
print(len(lat_v), max(lat_v), min(lat_v))
```

```
106 45.98 41.78
```

```
In [17]: n= 0
for i in range(int(min(lat_v)*100), int((max(lat_v)+0.04)*100), 4):
    n= n+1
    print(n)
    print(i/100)
```

1
41.78
2
41.82
3
41.86
4
41.9
5
41.94
6
41.98
7
42.02
8
42.06
9
42.1
10
42.14
11
42.18
12
42.22
13
42.26
14
42.3
15
42.34
16
42.38
17
42.42
18
42.46
19
42.5
20
42.54
21
42.58
22
42.62
23
42.66
24
42.7
25
42.74
26
42.78
27
42.82
28
42.86

29
42.9
30
42.94
31
42.98
32
43.02
33
43.06
34
43.1
35
43.14
36
43.18
37
43.22
38
43.26
39
43.3
40
43.34
41
43.38
42
43.42
43
43.46
44
43.5
45
43.54
46
43.58
47
43.62
48
43.66
49
43.7
50
43.74
51
43.78
52
43.82
53
43.86
54
43.9
55
43.94
56
43.98

57
44.02
58
44.06
59
44.1
60
44.14
61
44.18
62
44.22
63
44.26
64
44.3
65
44.34
66
44.38
67
44.42
68
44.46
69
44.5
70
44.54
71
44.58
72
44.62
73
44.66
74
44.7
75
44.74
76
44.78
77
44.82
78
44.86
79
44.9
80
44.94
81
44.98
82
45.02
83
45.06
84
45.1

85
45.14
86
45.18
87
45.22
88
45.26
89
45.3
90
45.34
91
45.38
92
45.42
93
45.46
94
45.5
95
45.54
96
45.58
97
45.62
98
45.66
99
45.7
100
45.74
101
45.78
102
45.82
103
45.86
104
45.9
105
45.94
106
45.98

```
In [18]: np.shape(dataset)
```

```
Out[18]: (106, 79)
```

```
In [19]: n = 0  
for i in range(int(min(lon_v)*100), int((max(lon_v)+0.04)*100), 4):  
    n=n+1  
    print(n)  
    print(i)
```

1
-8790
2
-8786
3
-8782
4
-8778
5
-8774
6
-8770
7
-8766
8
-8762
9
-8758
10
-8754
11
-8750
12
-8746
13
-8742
14
-8738
15
-8734
16
-8730
17
-8726
18
-8722
19
-8718
20
-8714
21
-8710
22
-8706
23
-8702
24
-8698
25
-8694
26
-8690
27
-8686
28
-8682

29
-8678
30
-8674
31
-8670
32
-8666
33
-8662
34
-8658
35
-8654
36
-8650
37
-8646
38
-8642
39
-8638
40
-8634
41
-8630
42
-8626
43
-8622
44
-8618
45
-8614
46
-8610
47
-8606
48
-8602
49
-8598
50
-8594
51
-8590
52
-8586
53
-8582
54
-8578
55
-8574
56
-8570

57
-8566
58
-8562
59
-8558
60
-8554
61
-8550
62
-8546
63
-8542
64
-8538
65
-8534
66
-8530
67
-8526
68
-8522
69
-8518
70
-8514
71
-8510
72
-8506
73
-8502
74
-8498
75
-8494
76
-8490
77
-8486
78
-8482
79
-8478

Experiment:

```
In [20]: # Count non-nan  
non_nan_num = dataset.count().sum()  
non_nan_num
```

Out [20]: 3599

```
In [21]: cloud_pixel_num = (dataset>0.2).sum().sum()  
cloud_pixel_num
```

Out [21]: 3072

[Q] Is it a good idea to consider values in a list that are smaller than the mean value minus standard deviation as noise?

[A] It depends on the context and the distribution of the values in the list. In general, the mean value minus one standard deviation is used to define the range within which approximately 68% of the values in a normally distributed data set fall. Therefore, values that fall outside this range could be considered outliers or potentially noise. However, if the distribution of the data is skewed or has a long tail, this may not be a good approach as it may incorrectly identify values as noise. Additionally, the decision of whether to consider values as noise depends on the specific analysis being conducted and the goals of the analysis.

```
In [22]: mean = dataset.mean(numeric_only=True, skipna=True).mean()  
std = dataset.stack().std(skipna=True)  
print("Mean:", mean)  
print("Standard deviation:", std)
```

Mean: 0.36573900893697614
Standard deviation: 0.13373368120646276

```
In [23]: i_testing = dataset.to_numpy().ravel()  
i_2d = dataset.values
```

```
In [24]: x_testing = np.array(i_testing).reshape(-1, 1)  
# identify NaN values  
mask = np.isnan(x_testing)  
  
# create new array without NaN values  
x_testing = x_testing[~mask]  
x_testing = np.array(x_testing).reshape(-1, 1)  
x_testing
```

```
Out [24]: array([[0.5725   ],  
                [0.5275   ],  
                [0.6475   ],  
                ...,  
                [0.385    ],  
                [0.375    ],  
                [0.35999998]])
```

```
In [25]: import plotly.express as px  
from sklearn.mixture import GaussianMixture  
from scipy import stats  
from scipy.optimize import fsolve  
from scipy.stats import norm
```

```

import plotly

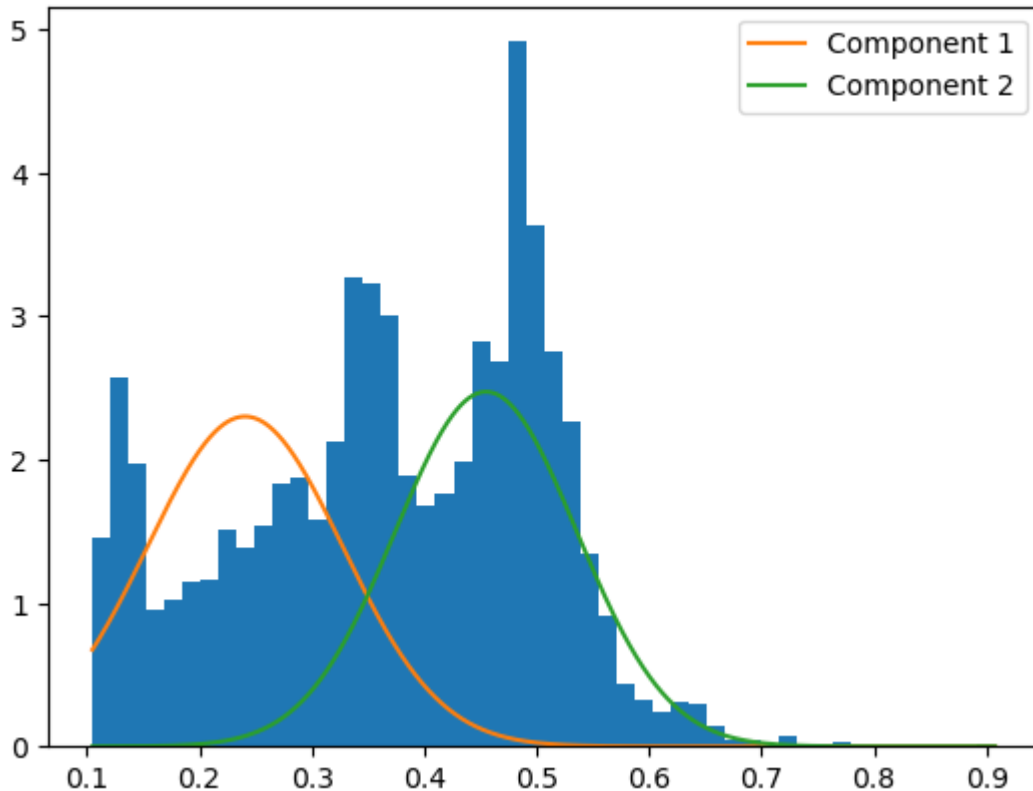
import matplotlib.pyplot as plt
import matplotlib.cm as cm

# Fit a mixture of two Gaussians to the data
gmm_11 = GaussianMixture(n_components=2).fit(x_testing)

# Get the means and standard deviations of the two Gaussian components
mu1_11, mu2_11 = gmm_11.means_.flatten()
sigma1_11, sigma2_11 = np.sqrt(gmm_11.covariances_.flatten())

# Visualize the results
plt.hist(i_testing, density=True, bins = 50)
# x_axis = i_11.unique()
x_axis_11 = np.linspace(min(x_testing), max(x_testing), 3000)
plt.plot(x_axis_11, 0.5 * np.exp(-(x_axis_11 - mu1_11)**2 / (2 * sigma1_11**2)))
plt.plot(x_axis_11, 0.5 * np.exp(-(x_axis_11 - mu2_11)**2 / (2 * sigma2_11**2)))
plt.legend()
plt.show()

```



```

In [26]: # Define the two normal distributions
mean1, std1 = mu1_11, sigma1_11
mean2, std2 = mu2_11, sigma2_11

dist1 = norm(mean1, std1)
dist2 = norm(mean2, std2)

# Define a function to find the difference between the two distributions
def diff(x):
    return dist1.pdf(x) - dist2.pdf(x)

```

```
# Use fsolve to find the x value where the difference is zero
x_cross_11 = fsolve(diff, x0=(mean1+mean2)/2)

print("Cross point of two normal distributions: x =", x_cross_11[0])
```

Cross point of two normal distributions: x = 0.349343237683899

In []:

In []:

In []:

In []:

END OF EXPERIMENT

```
In [27]: def image_statistics(Z):
#Input: Z, a 2D array, hopefully containing some sort of peak
#Output: cx,cy,sx,sy,skx,sky,kx,ky
#cx and cy are the coordinates of the centroid
#sx and sy are the standard deviation in the x and y directions
#skx and sky are the skewness in the x and y directions
#kx and ky are the Kurtosis in the x and y directions
#Note: this is not the excess kurtosis. For a normal distribution
#you expect the kurtosis will be 3.0. Just subtract 3 to get the
#excess kurtosis.
    import numpy as np

    h,w = np.shape(Z)

    x = range(w) = 79
    y = range(h) = 106
    x = range(int(min(lon_v)*100), int((max(lon_v)+0.04)*100), 4) # 79 (-847
    y = range(int(min(lat_v)*100), int((max(lat_v)+0.04)*100), 4) # 106 (459

    Z_flatten = Z.to_numpy().reshape(8374, 1)
    i_testing = Z.to_numpy().ravel()
    i_2d = Z.values

    x_testing = np.array(i_testing).reshape(-1, 1)
    # identify NaN values
    mask = np.isnan(x_testing)

    # create new array without NaN values
    x_testing = x_testing[~mask]
    x_testing = np.array(x_testing).reshape(-1, 1)

    # Fit a mixture of two Gaussians to the data
    gmm_11 = GaussianMixture(n_components=2).fit(x_testing)

    # Get the means and standard deviations of the two Gaussian components
```

```

#     mu1_11, mu2_11 = gmm_11.means_.flatten()
#     sigma1_11, sigma2_11 = np.sqrt(gmm_11.covariances_.flatten())

#     # Define the two normal distributions
#     mean1, std1 = mu1_11, sigma1_11
#     mean2, std2 = mu2_11, sigma2_11

#     dist1 = norm(mean1, std1)
#     dist2 = norm(mean2, std2)

#     # Use fsolve to find the x value where the difference is zero
#     x_cross_11 = fsolve(diff, x0=(mean1+mean2)/2)
cutt_off_point = 0.1

# Count non-nan
non_nan_num = Z.count().sum()

mean = Z.mean(numeric_only=True, skipna=True).mean()
std = Z.stack().std(skipna=True)

temp_thres = mean - std

#     q1 = Z.stack().quantile(0.2)

#     if cutt_off_point > 0:
#         threshold = cutt_off_point
#     else:
#         threshold = q1

cloud_pixel_num = (Z>threshold).sum().sum()

percentage_cloud_cover = cloud_pixel_num/non_nan_num

#calculate projections along the x and y axes
yp = np.sum(Z,axis=1) # column # 79
xp = np.sum(Z,axis=0) # row # 106

#centroid
cx = np.sum(x*xp/100)/np.sum(xp)
cy = np.sum(y*yp/100)/np.sum(yp)

#standard deviation
x2 = ((x-(cx * 100))/100)**2
y2 = ((y-(cy*100))/100)**2

sx = np.sqrt( np.sum(x2*xp)/np.sum(xp) )
sy = np.sqrt( np.sum(y2*yp)/np.sum(yp) )

#skewness
x3 = ((x-(cx * 100))/100)**3
y3 = ((y-(cy*100))/100)**3

```

```

skx = np.sum(xp*x3)/(np.sum(xp) * sx**3)
sky = np.sum(yp*y3)/(np.sum(yp) * sy**3)

#Kurtosis
x4 = ((x-(cx * 100))/100)**4
y4 = ((y-(cy*100))/100)**4
kx = np.sum(xp*x4)/(np.sum(xp) * sx**4)
ky = np.sum(yp*y4)/(np.sum(yp) * sy**4)

return cx,cy,sx,sy,skx,sky,kx,ky,cloud_pixel_num,non_nan_num,percentage_

```

```
In [28]: image_statistics(dataset)
```

```

Out[28]: (-86.81064514778352,
43.6139104387241,
0.6474604854343116,
1.2508422288024557,
0.7469282114823914,
0.44760817009606163,
3.1520023667382913,
1.8988711406601753,
3599,
3599,
1.0,
0.1,
array([nan, nan, nan, ..., nan, nan, nan]),
array([[nan, nan, nan, ..., nan, nan, nan],
       [nan, nan, nan, ..., nan, nan, nan],
       [nan, nan, nan, ..., nan, nan, nan],
       ...,
       [nan, nan, nan, ..., nan, nan, nan],
       [nan, nan, nan, ..., nan, nan, nan],
       [nan, nan, nan, ..., nan, nan, nan]]))

```

```

In [29]: for fn in filename[:2]:
          print(int(fn[9:19].replace('.', '')))
          print('{value}'.format(value=fn[20:24]))

```

```

20131001
0000
20131001
0030

```

```

In [30]: list_da = []
list_ti = []
list_m = []
list_cx = []
list_cy = []
list_sx = []
list_sy = []
list_skx = []
list_sky = []
list_kx = []
list_ky = []
list_cloud_pixel_num = []

```

```
list_non_nan_num = []
list_percentage_cloud_cover = []
list_threshold = []
list_1d = []
list_2d = []
```

```
In [31]: import warnings
warnings.filterwarnings('ignore')

for fn in tqdm(filename):
#     print('{value}'.format(value=fn[20:24]))
    file = pd.read_csv(fn, index_col = 0)
    #print(file.shape)

    date_f = int(fn[9:19].replace('.', ''))
    list_da.append(date_f)

    time_f = '{value}'.format(value=fn[20:24])
    list_ti.append(time_f)

    mean_f = file.mean().mean()
    list_m.append(mean_f)

    cx_f = image_statistics(file)[0]
    list_cx.append(cx_f)

    cy_f = image_statistics(file)[1]
    list_cy.append(cy_f)

    sx_f = image_statistics(file)[2]
    list_sx.append(sx_f)

    sy_f = image_statistics(file)[3]
    list_sy.append(sy_f)

    skx_f = image_statistics(file)[4]
    list_skx.append(skx_f)

    sky_f = image_statistics(file)[5]
    list_sky.append(sky_f)

    kx_f = image_statistics(file)[6]
    list_kx.append(kx_f)

    ky_f = image_statistics(file)[7]
    list_ky.append(ky_f)

    cloud_pixel_num_f = image_statistics(file)[8]
    list_cloud_pixel_num.append(cloud_pixel_num_f)

    non_nan_num_f = image_statistics(file)[9]
    list_non_nan_num.append(non_nan_num_f)

    percentage_cloud_cover_f = image_statistics(file)[10]
    list_percentage_cloud_cover.append(percentage_cloud_cover_f)
```



```
threshold_f = image_statistics(file)[11]
list_threshold.append(threshold_f)

f1d_f = image_statistics(file)[12]
list_1d.append(f1d_f)

f2d_f = image_statistics(file)[13]
list_2d.append(f2d_f)
```

100%|██████████| 15607/15607 [21:09<00:00, 12.30it/s]

In [32]: `print(len(list_m))`

15607

In [33]: `print(list_ti)`

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

'0','1145','1200','1230','1245','1300','1315','1330','1345','1400'
'0','1415','1430','1445','1500','1530','1545','1600','1615','1630',
'1645','1700','1715','1730','1745','1800','1830','1845','1900','19
15','1930','1945','2000','2015','2030','2100','2130','2145','220
0','2215','2230','2245','2300','2315','2330','2345','0000','0030',
'0100','0115','0130','0145','0200','0215','0230','0245','0300','03
30','0345','0400','0415','0430','0445','0500','0515','0530','054
5','0600','0630','0645','0700','0715','0730','0745','0800','0815',
'0830','0845','0900','0930','0945','1000','1015','1030','1045','11
00','1115','1130','1145','1200','1230','1245','1300','1315','133
0','1345','1400','1415','1430','1445','1500','1530','1545','1600',
'1615','1630','1645','1700','1715','1730','1745','1800','1830','18
45','1900','1915','1930','1945','2015','2030','2100','2130','214
5','2200','2215','2230','2245','2300','2315','2330','2345','0000',
'0030','0100','0115','0130','0145','0200','0215','0230','0245','03
00','0330','0345','0400','0415','0430','0445','0500','0515','053
0','0545','0600','0630','0645','0700','0715','0730','0745','0800',
'0815','0830','0845','0900','0930','0945','1000','1015','1030','10
45','1100','1115','1130','1145','1200','1230','1245','1300','131
5','1330','1345','1400','1415','1430','1445','1500','1530','1545',
'1600','1615','1630','1645','1700','1715','1730','1745','1800','18
30','1845','1900','1915','1930','1945','2000','2015','2030','210
0','2130','2145','2200','2215','2230','2245','2300','2315','2330',
'2345','0000','0030','0100','0115','0130','0145','0200','0215','02
30','0245','0300','0330','0345','0400','0415','0430','0445','050
0','0515','0530','0545','0600','0630','0645','0700','0715','0730',
'0745','0800','0815','0830','0845','0900','0930','0945','1000','10
15','1030','1045','1100','1115','1130','1145','1200','1230','124
5','1300','1315','1330','1345','1400','1415','1430','1445','1500',
'1530','1545','1600','1615','1630','1645','1700','1715','1730','17
45','1800','1830','1845','1900','1915','1930','1945','2000','201
5','2030','2100','2130','2145','2200','2215','2230','2245','2300',
'2315','2330','2345','0000','0030','0100','0115','0130','0145','02
00','0215','0230','0245','0300','0330','0345','0400','0415','043
0','0445','0500','0515','0530','0545','0600','0630','0645','0700',
'0715','0730','0745','0800','0815','0830','0845','0900','0930','09
45','1000','1015','1030','1045','1100','1115','1130','1145','120
0','1230','1245','1300','1315','1330','1345','1400','1415','1430',
'1445','1500','1530','1545','1600','1615','1630','1645','1700','17
15','1730','1745','1800','1830','1845','1900','1915','1930','194
5','2000','2015','2030','2100','2130','2145','2200','2215','2230',
'2245','2300','2315','2330','2345','0000','0030','0100','0115','01
30','0145','0200','0215','0230','0245','0300','0330','0345','040
0','0415','0430','0445','0500','0515','0530','0545','0600','0630',
'0645','0700','0715','0730','0745','0800','0815','0830','0845','09
00','0930','0945','1000','1015','1030','1045','1100','1115','113
0','1145','1200','1230','1245','1300','1315','1330','1345','1400',
'1415','1430','1445','1500','1530','1545','1600','1615','1630','16
45','1700','1715','1730','1745','1800','1830','1845','1900','191
5','1930','1945','2000','2015','2030','2100','2130','2145','2200',
'2215','2230','2245','2300','2315','2330','2345','0000','0030','01
00','0115','0130','0145','0200','0215','0230','0245','0300','033
0','0345','0400','0415','0430','0445','0500','0515','0530','0545',
'0600','0630','0645','0700','0715','0730','0745','0800','0815','08
30','0845','0900','0930','0945','1000','1015','1030','1045','110
0','1115','1130','1145','1200','1230','1245','1300','1315','1330'

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

'1015', '1030', '1045', '1060', '1075', '1090', '1105', '1120', '1135', '1150', '1165', '1180', '1195', '1210', '1225', '1240', '1255', '1270', '1285', '1300', '1315', '1330', '1345', '1360', '1375', '1390', '1405', '1420', '1435', '1450', '1465', '1480', '1495', '1510', '1525', '1540', '1555', '1570', '1585', '1600', '1615', '1630', '1645', '1660', '1675', '1690', '1705', '1720', '1735', '1750', '1765', '1780', '1795', '1810', '1825', '1840', '1855', '1870', '1885', '1900', '1915', '1930', '1945', '1960', '1975', '1990', '2005', '2020', '2035', '2050', '2065', '2080', '2095', '2110', '2125', '2140', '2155', '2170', '2185', '2200', '2215', '2230', '2245', '2260', '2275', '2290', '2305', '2320', '2335', '2350', '2365', '2380', '2395', '2410', '2425', '2440', '2455', '2470', '2485', '2500', '2515', '2530', '2545', '2560', '2575', '2590', '2605', '2620', '2635', '2650', '2665', '2680', '2695', '2710', '2725', '2740', '2755', '2770', '2785', '2800', '2815', '2830', '2845', '2860', '2875', '2890', '2905', '2920', '2935', '2950', '2965', '2980', '2995', '3010', '3025', '3040', '3055', '3070', '3085', '3100', '3115', '3130', '3145', '3160', '3175', '3190', '3205', '3220', '3235', '3250', '3265', '3280', '3295', '3310', '3325', '3340', '3355', '3370', '3385', '3400', '3415', '3430', '3445', '3460', '3475', '3490', '3505', '3520', '3535', '3550', '3565', '3580', '3595', '3610', '3625', '3640', '3655', '3670', '3685', '3700', '3715', '3730', '3745', '3760', '3775', '3790', '3805', '3820', '3835', '3850', '3865', '3880', '3895', '3910', '3925', '3940', '3955', '3970', '3985', '4000', '4015', '4030', '4045', '4060', '4075', '4090', '4105', '4120', '4135', '4150', '4165', '4180', '4195', '4210', '4225', '4240', '4255', '4270', '4285', '4300', '4315', '4330', '4345', '4360', '4375', '4390', '4405', '4420', '4435', '4450', '4465', '4480', '4495', '4510', '4525', '4540', '4555', '4570', '4585', '4600', '4615', '4630', '4645', '4660', '4675', '4690', '4705', '4720', '4735', '4750', '4765', '4780', '4795', '4810', '4825', '4840', '4855', '4870', '4885', '4900', '4915', '4930', '4945', '4960', '4975', '4990', '5005', '5020', '5035', '5050', '5065', '5080', '5095', '5110', '5125', '5140', '5155', '5170', '5185', '5200', '5215', '5230', '5245', '5260', '5275', '5290', '5305', '5320', '5335', '5350', '5365', '5380', '5395', '5410', '5425', '5440', '5455', '5470', '5485', '5500', '5515', '5530', '5545', '5560', '5575', '5590', '5605', '5620', '5635', '5650', '5665', '5680', '5695', '5710', '5725', '5740', '5755', '5770', '5785', '5800', '5815', '5830', '5845', '5860', '5875', '5890', '5905', '5920', '5935', '5950', '5965', '5980', '5995', '6010', '6025', '6040', '6055', '6070', '6085', '6100', '6115', '6130', '6145', '6160', '6175', '6190', '6205', '6220', '6235', '6250', '6265', '6280', '6295', '6310', '6325', '6340', '6355', '6370', '6385', '6400', '6415', '6430', '6445', '6460', '6475', '6490', '6505', '6520', '6535', '6550', '6565', '6580', '6595', '6610', '6625', '6640', '6655', '6670', '6685', '6700', '6715', '6730', '6745', '6760', '6775', '6790', '6805', '6820', '6835', '6850', '6865', '6880', '6895', '6910', '6925', '6940', '6955', '6970', '6985', '7000', '7015', '7030', '7045', '7060', '7075', '7090', '7105', '7120', '7135', '7150', '7165', '7180', '7195', '7210', '7225', '7240', '7255', '7270', '7285', '7300', '7315', '7330', '7345', '7360', '7375', '7390', '7405', '7420', '7435', '7450', '7465', '7480', '7495', '7510', '7525', '7540', '7555', '7570', '7585', '7600', '7615', '7630', '7645', '7660', '7675', '7690', '7705', '7720', '7735', '7750', '7765', '7780', '7795', '7810', '7825', '7840', '7855', '7870', '7885', '7900', '7915', '7930', '7945', '7960', '7975', '7990', '8005', '8020', '8035', '8050', '8065', '8080', '8095', '8110', '8125', '8140', '8155', '8170', '8185', '8200', '8215', '8230', '8245', '8260', '8275', '8290', '8305', '8320', '8335', '8350', '8365', '8380', '8395', '8410', '8425', '8440', '8455', '8470', '8485', '8500', '8515', '8530', '8545', '8560', '8575', '8590', '8605', '8620', '8635', '8650', '8665', '8680', '8695', '8710', '8725', '8740', '8755', '8770', '8785', '8800', '8815', '8830', '8845', '8860', '8875', '8890', '8905', '8920', '8935', '8950', '8965', '8980', '8995', '9010', '9025', '9040', '9055', '9070', '9085', '9100', '9115', '9130', '9145', '9160', '9175', '9190', '9205', '9220', '9235', '9250', '9265', '9280', '9295', '9310', '9325', '9340', '9355', '9370', '9385', '9400', '9415', '9430', '9445', '9460', '9475', '9490', '9505', '9520', '9535', '9550', '9565', '9580', '9595', '9610', '9625', '9640', '9655', '9670', '9685', '9700', '9715', '9730', '9745', '9760', '9775', '9790', '9805', '9820', '9835', '9850', '9865', '9880', '9895', '9910', '9925', '9940', '9955', '9970', '9985', '10000', '10005', '10010', '10015', '10020', '10025', '10030', '10035', '10040', '10045', '10050', '10055', '10060', '10065', '10070', '10075', '10080', '10085', '10090', '10095', '10100', '10105', '10110', '10115', '10120', '10125', '10130', '10135', '10140', '10145', '10150', '10155', '10160', '10165', '10170', '10175', '10180', '10185', '10190', '10195', '10200', '10205', '10210', '10215', '10220', '10225', '10230', '10235', '10240', '10245', '10250', '10255', '10260', '10265', '10270', '10275', '10280', '10285', '10290', '10295', '10300', '10305', '10310', '10315', '10320', '10325', '10330', '10335', '10340', '10345', '10350', '103

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

'0000','0030','0100','0115','0130','0145','0200','0215','0230','
'0245','0300','0330','0345','0400','0415','0430','0445','0500','05
15','0530','0545','0600','0630','0645','0700','0715','0730','074
5','0800','0815','0830','0845','0900','0930','0945','1000','1015',
'1030','1045','1100','1115','1130','1145','1200','1230','1245','13
00','1315','1330','1345','1400','1415','1430','1445','1500','153
0','1545','1600','1615','1630','1645','1700','1715','1730','1745',
'1800','1830','1845','1900','1915','1930','1945','2000','2015','20
30','2100','2130','2145','2200','2215','2230','2245','2300','231
5','2330','2345','0000','0030','0100','0115','0130','0145','0200',
'0215','0230','0245','0300','0330','0345','0400','0415','0430','04
45','0500','0515','0530','0545','0600','0630','0645','0700','071
5','0730','0745','0800','0815','0830','0845','0900','0930','0945',
'1000','1015','1030','1045','1100','1115','1130','1145','1200','12
30','1245','1300','1315','1330','1345','1400','1415','1430','144
5','1500','1530','1545','1600','1615','1630','1645','1700','1715',
'1730','1745','1800','1830','1845','1900','1915','1930','1945','20
00','2015','2030','2100','2130','2145','2200','2215','2230','224
5','2300','2315','2330','2345','0000','0030','0100','0115','0130',
'0145','0200','0215','0230','0245','0300','0330','0345','0400','04
15','0430','0445','0500','0515','0530','0545','0600','0630','064
5','0700','0715','0730','0745','0800','0815','0830','0845','0900',
'0930','0945','1000','1015','1030','1045','1100','1115','1130','11
45','1200','1230','1245','1300','1315','1330','1345','1400','141
5','1430','1445','1500','1530','1545','1600','1615','1630','1645',
'1700','1715','1730','1745','1800','1830','1845','1900','1915','19
30','1945','2015','2030','2100','2130','2145','2200','2215','223
0','2245','2300','2315','2330','2345','0000','0030','0100','0115',
'0130','0145','0200','0215','0230','0245','0300','0330','0345','04
00','0415','0430','0445','0500','0515','0530','0545','0600','063
0','0645','0700','0715','0730','0745','0800','0815','0830','0845',
'0900','0930','0945','1000','1015','1030','1045','1100','1115','11
30','1145','1200','1230','1245','1300','1315','1330','1345','140
0','1415','1430','1445','1500','1530','1545','1600','1615','1630',
'1645','1700','1715','1730','1745','1800','1830','1845','1900','19
15','1930','1945','2000','2015','2030','2100','2130','2145','220
0','2215','2230','2245','2300','2315','2330','2345','0000','0030',
'0100','0115','0130','0145','0200','0215','0230','0245','0300','03
30','0345','0400','0415','0430','0445','0500','0515','0530','054
5','0600','0630','0645','0700','0715','0730','0745','0800','0815',
'0830','0845','0900','0930','0945','1000','1015','1030','1045','11
00','1115','1130','1145','1200','1230','1245','1300','1315','133
0','1345','1400','1415','1430','1445','1500','1530','1545','1600',
'1615','1630','1645','1700','1715','1730','1745','1800','1830','18
45','1900','1915','1930','1945','2000','2015','2030','2100','213
0','2145','2200','2215','2230','2245','2300','2315','2330','2345',
'0000','0030','0100','0115','0130','0145','0200','0215','0230','02
45','0300','0330','0345','0400','0415','0430','0445','0500','051
5','0530','0545','0600','0630','0645','0700','0715','0730','0745',
'0800','0815','0830','0845','0900','0930','0945','1000','1015','10
30','1045','1100','1115','1130','1145','1200','1230','1245','130
0','1315','1330','1345','1400','1415','1430','1445','1500','1530',
'1545','1600','1615','1630','1645','1700','1715','1730','1745','18
00','1830','1845','1900','1915','1930','1945','2000','2015','203
0','2100','2130','2145','2200','2215','2230','2245','2300','2315',
'2330','2345','0000','0030','0100','0115','0130','0145','0200','02

[illegible]

[illegible]

'0800','0815','0830','0845','0900','0930','0945','1000','1015','
'1030','1045','1100','1115','1130','1145','1200','1230','1245','13
00','1315','1330','1345','1400','1415','1430','1445','1500','153
0','1545','1600','1615','1630','1645','1700','1715','1730','1745',
'1800','1830','1845','1900','1915','1930','1945','2000','2015','20
30','2100','2130','2145','2200','2215','2230','2245','2300','231
5','2330','2345','0000','0030','0100','0115','0130','0145','0200',
'0215','0230','0245','0300','0330','0345','0400','0415','0430','04
45','0500','0515','0530','0545','0600','0630','0645','0700','071
5','0730','0745','0800','0815','0830','0845','0900','0930','0945',
'1000','1015','1030','1045','1100','1115','1130','1145','1200','12
30','1245','1300','1315','1330','1345','1400','1415','1430','144
5','1500','1530','1545','1600','1615','1630','1645','1700','1715',
'1730','1745','1800','1830','1845','1900','1915','1930','1945','20
00','2015','2030','2100','2145','2200','2215','2230','2245','230
0','2315','2330','2345','0000','0030','0100','0115','0130','0145',
'0200','0215','0230','0245','0300','0330','0345','0400','0415','04
30','0445','0500','0515','0530','0545','0600','0630','0645','070
0','0715','0730','0745','0800','0815','0830','0845','0900','0930',
'0945','1000','1015','1030','1045','1100','1115','1130','1145','12
00','1230','1245','1300','1315','1330','1345','1400','1415','143
0','1445','1500','1530','1545','1600','1615','1630','1645','1700',
'1715','1730','1745','1800','1830','1845','1900','1915','1930','19
45','2000','2015','2030','2100','2130','2145','2200','2215','223
0','2245','2300','2315','2330','2345','0000','0030','0100','0115',
'0130','0145','0200','0215','0230','0245','0300','0330','0345','04
00','0415','0430','0445','0500','0515','0530','0545','0600','063
0','0645','0700','0715','0730','0745','0800','0815','0830','0845',
'0900','0930','0945','1000','1015','1030','1045','1100','1115','11
30','1145','1200','1230','1245','1300','1315','1330','1345','140
0','1415','1430','1445','1500','1530','1545','1600','1615','1630',
'1645','1700','1715','1730','1745','1800','1830','1845','1900','19
15','1930','1945','2000','2015','2030','2100','2145','2200','221
5','2230','2245','2300','2315','2330','2345','0000','0030','0100',
'0115','0130','0145','0200','0215','0230','0245','0300','0330','03
45','0400','0415','0430','0445','0500','0515','0530','0545','060
0','0630','0645','0700','0715','0730','0745','0800','0815','0830',
'0845','0900','0930','0945','1000','1015','1030','1045','1100','11
15','1130','1145','1200','1230','1245','1300','1315','1330','134
5','1400','1415','1430','1445','1500','1530','1545','1600','1615',
'1630','1645','1700','1715','1730','1745','1800','1830','1845','19
00','1915','1930','1945','2000','2015','2030','2100','2145','220
0','2215','2230','2245','2300','2315','2330','2345','0000','0030',
'0100','0115','0130','0145','0200','0215','0230','0245','0300','03
30','0345','0400','0415','0430','0445','0500','0515','0530','054
5','0600','0630','0645','0700','0715','0730','0745','0800','0815',
'0830','0845','0900','0930','0945','1000','1015','1030','1045','11
00','1115','1130','1145','1200','1230','1245','1300','1315','133
0','1345','1400','1415','1430','1445','1500','1530','1545','1600',
'1615','1630','1645','1700','1715','1730','1745','1800','1830','18
45','1900','1915','1930','1945','2015','2030','2100','2130','214
5','2200','2215','2230','2245','2300','2315','2330','2345','0000',
'0030','0100','0115','0130','0145','0200','0215','0230','0245','03
00','0330','0345','0400','0415','0430','0445','0500','0515','053
0','0545','0600','0630','0645','0700','0715','0730','0745','0800',
'0815','0830','0845','0900','0930','0945','1000','1015','1030','10

[illegible]

[illegible]

[illegible]

```

15', '1730', '1745', '1800', '1830', '1845', '1900', '1915', '1930', '194
5', '2000', '2015', '2030', '2100', '2130', '2145', '2200', '2215', '2230',
'2245', '2300', '2315', '2330', '2345', '0000', '0030', '0100', '0115', '01
30', '0145', '0200', '0215', '0230', '0245', '0300', '0330', '0345', '040
0', '0415', '0430', '0445', '0500', '0515', '0530', '0545', '0600', '0630',
'0645', '0700', '0715', '0730', '0745', '0800', '0815', '0845', '0900', '09
30', '0945', '1000', '1015', '1030', '1045', '1100', '1115', '1130', '114
5', '1200', '1230', '1245', '1300', '1315', '1330', '1345', '1400', '1415',
'1430', '1445', '1500', '1530', '1545', '1600', '1615', '1630', '1645', '17
00', '1715', '1730', '1745', '1800', '1830', '1845', '1900', '1915', '193
0', '1945', '2000', '2015', '2030', '2100', '2130', '2145', '2200', '2215',
'2230', '2245', '2300', '2315', '2330', '2345', '0000', '0030', '0045', '01
00', '0115', '0130', '0145', '0200', '0215', '0230', '0245', '0300', '033
0', '0345', '0400', '0415', '0430', '0445', '0500', '0515', '0530', '0545',
'0600', '0630', '0645', '0700', '0715', '0730', '0745', '0800', '0815', '08
30', '0845', '0900', '0930', '0945', '1000', '1015', '1030', '1045', '110
0', '1115', '1130', '1145', '1200', '1230', '1245', '1300', '1315', '1330',
'1345', '1400', '1415', '1430', '1445', '1500', '1530', '1545', '1600', '16
15', '1630', '1645', '1700', '1715', '1730', '1745', '1800', '1830', '184
5', '1900', '1915', '1930', '1945', '2015', '2030', '2100', '2130', '2145',
'2200', '2215', '2230', '2245', '2300', '2315', '2330', '2345', '0000', '00
30', '0100', '0115', '0130', '0145', '0200', '0215', '0230', '0245', '030
0', '0330', '0345', '0400', '0415', '0430', '0445', '0500', '0515', '0530',
'0545', '0600', '0630', '0645', '0700', '0715', '0730', '0745', '0800', '08
15', '0830', '0845', '0900', '0930', '0945', '1000', '1015', '1030', '104
5', '1100', '1115', '1130', '1145', '1200', '1230', '1245', '1300', '1315',
'1330', '1345', '1400', '1415', '1430', '1445', '1500', '1530', '1545', '16
00', '1615', '1630', '1645', '1700', '1715', '1730', '1745', '1800', '183
0', '1845', '1900', '1915', '1930', '1945', '2000', '2015', '2030', '2100',
'2130', '2145', '2200', '2215', '2230', '2245', '2300', '2315', '2330', '23
45', '0000', '0030', '0100', '0115', '0130', '0145', '0200', '0215', '023
0', '0245', '0300', '0330', '0345', '0400', '0415', '0430', '0445', '0500',
'0515', '0530', '0545', '0600', '0630', '0645', '0700', '0715', '0730', '07
45', '0800', '0815', '0830', '0845', '0900', '0930', '0945', '1000', '101
5', '1030', '1045', '1100', '1115', '1130', '1145', '1200', '1230', '1245',
'1300', '1315', '1330', '1345', '1400', '1415', '1430', '1445', '1500', '15
30', '1545', '1600', '1615', '1630', '1645', '1700', '1715', '1730', '174
5', '1800', '1830', '1845', '1900', '1915', '1930', '1945', '2000', '2015',
'2030', '2100', '2130', '2145', '2200', '2215', '2230', '2245', '2300', '23
15', '2330', '2345', '0000', '0030', '0100', '0115', '0130', '0145', '020
0', '0215', '0230', '0245', '0300', '0330', '0345', '0400', '0415', '0430',
'0445', '0500', '0515', '0530', '0545', '0600', '0630', '0645', '0700', '07
15', '0730', '0745', '0800', '0815', '0830', '0845', '0900', '0930', '094
5', '1000', '1015', '1030', '1045', '1100', '1115', '1130', '1145', '1200',
'1230', '1245', '1300', '1315', '1330', '1345', '1400', '1415', '1430', '14
45', '1500', '1530', '1545', '1600', '1615', '1630', '1645', '1700', '171
5', '1730', '1745', '1800', '1830', '1845', '1900', '1915', '1930', '1945',
'2000', '2015', '2030', '2100', '2130', '2145', '2200', '2215', '2230', '22
45', '2300', '2315', '2330', '234
5']

```

In [34]: `import csv`

TO-DO:

Make sure to change the directory for the outputs, and ensure that the file is labeled properly as it is yielded from **Step 3** .

```
In [35]: #columns = ['Date','Time' 'Mean', 'Std', 'Skewness']
data = {'Date':list_da, 'Time': list_ti,'Mean':list_m, 'Centroid_lon':list_c
        'Std_lon':list_sx,'Std_lat':list_sy,'Skewness_lon':list_sxx, 'Skewne
        'Kurtosis_lon': list_kx, 'Kurtosis_lat': list_ky, 'Cloud_Cover_Count
        'Total_Lake_Pixel': list_non_nan_num, 'Percent_Cloud_Cover': list_pe
        'Cloud_Intensity_Threshold': list_threshold, 'Lake_1D': list_1d, 'La
df_1314fall = pd.DataFrame(data = data)

## T0-D0: Change directory for the outputs
df_1314fall.to_csv("/srv/scratch/N0AA/G0ES_Hourly_Statistics/stats_result/03
```

```
In [36]: df_1314fall.dtypes
```

```
Out[36]: Date                                int64
Time                                           object
Mean                                           float64
Centroid_lon                                float64
Centroid_lat                                float64
Std_lon                                       float64
Std_lat                                       float64
Skewness_lon                                float64
Skewness_lat                                float64
Kurtosis_lon                                float64
Kurtosis_lat                                float64
Cloud_Cover_Count                            int64
Total_Lake_Pixel                            int64
Percent_Cloud_Cover                         float64
Cloud_Intensity_Threshold                   float64
Lake_1D                                      object
Lake_2D                                      object
dtype: object
```

```
In [37]: df_1314fall
```


Out[37]:

	Date	Time	Mean	Centroid_Ion	Centroid_lat	Std_Ion	Std_lat	Skewnes
0	20131001	0000	0.002118	-86.758112	43.881262	0.651311	1.218475	0.66
1	20131001	0030	0.001995	-86.684720	44.006311	0.660303	1.245697	0.56
2	20131001	0100	0.001849	-86.761659	43.879946	0.627008	1.210121	0.61
3	20131001	0115	0.001956	-86.717053	43.837233	0.641098	1.285265	0.6
4	20131001	0130	0.002071	-86.731712	43.933932	0.658674	1.245903	0.65
...
15602	20140331	2245	0.266582	-86.718249	43.971807	0.641993	1.182278	0.6

	Date	Time	Mean	Centroid_Ion	Centroid_lat	Std_Ion	Std_lat	Skewnes
15603	20140331	2300	0.212935	-86.723271	43.961543	0.635073	1.168567	0.60

15604	20140331	2315	0.167716	-86.734206	43.904226	0.625233	1.175319	0.64
-------	----------	------	----------	------------	-----------	----------	----------	------

15605	20140331	2330	0.117806	-86.738856	43.914649	0.619529	1.170623	0.64
-------	----------	------	----------	------------	-----------	----------	----------	------

15606	20140331	2345	0.077016	-86.765821	43.811817	0.612643	1.189764	0.65
-------	----------	------	----------	------------	-----------	----------	----------	------

15607 rows × 17 columns

In []: