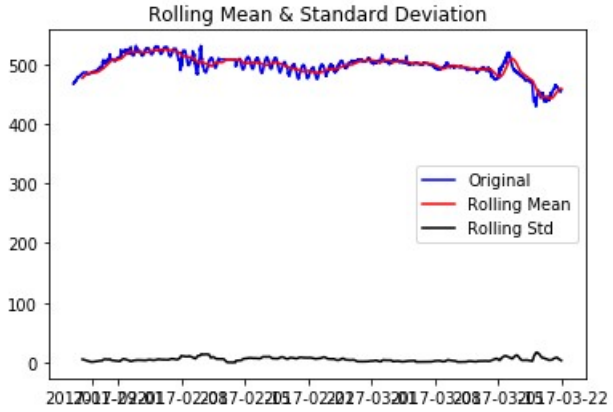
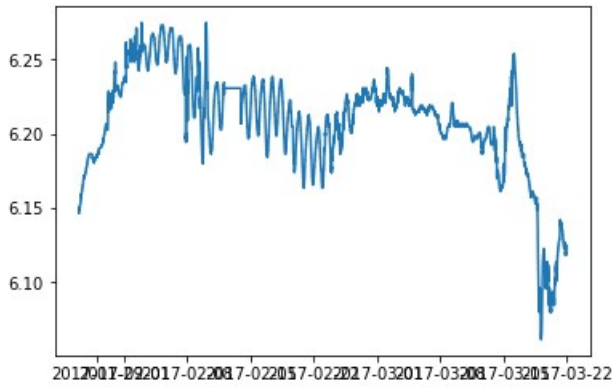
```
2017-03-21 18:15:00  5.045017
2017-03-21 18:30:00  4.949570
2017-03-21 18:45:00  4.872756
2017-03-21 19:00:00  4.811616
2017-03-21 19:15:00  4.748857
2017-03-21 19:30:00  4.697879
2017-03-21 19:45:00  4.609475
2017-03-21 20:00:00  4.569877
2017-03-21 20:15:00  4.401144
2017-03-21 20:30:00  4.271489
2017-03-21 20:45:00  4.126933
2017-03-21 21:00:00  4.003932
2017-03-21 21:15:00  3.875644
2017-03-21 21:30:00  3.769452
2017-03-21 21:45:00  3.696371
2017-03-21 22:00:00  3.609548
2017-03-21 22:15:00  3.543944
2017-03-21 22:30:00  3.476598
2017-03-21 22:45:00  3.397561
2017-03-21 23:00:00  3.338702
2017-03-21 23:15:00  3.298308
2017-03-21 23:30:00  3.249679
2017-03-21 23:45:00  3.217278

[5171 rows x 1 columns]
```

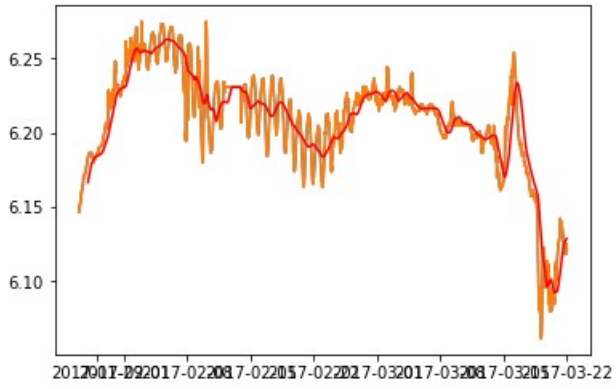
```
In [105]: orig = plt.plot(sc, color='blue', label='Original')
     ...: mean = plt.plot(sc_rolmean, color='red', label='Rolling Mean')
     ...: std = plt.plot(sc_rolstd, color='black', label='Rolling Std')
     ...: plt.legend(loc='best')
     ...: plt.title('Rolling Mean & Standard Deviation')
     ...: plt.show(block=False)
```



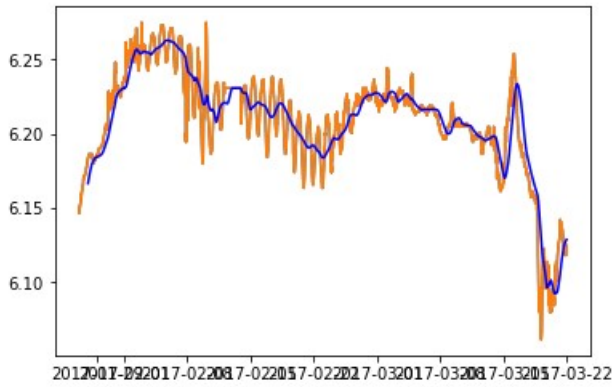

```
In [106]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
Out[106]: [<matplotlib.lines.Line2D at 0x2288faab278>]
```




```
In [107]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
     ...: #Determine rolling statistics
     ...: sc_moving_Average = sc_logScale.rolling(window=96).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=96).std()
     ...: plt.plot(sc_logScale)
     ...: plt.plot(sc_moving_Average, color='red')
     ...: #print(sc_rolmean,sc_rolstd)
Out[107]: [<matplotlib.lines.Line2D at 0x2288f2969e8>]
```




```
In [108]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
     ...: #Determine rolling statistics
     ...: sc_moving_Average = sc_logScale.rolling(window=96).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=96).std()
     ...: plt.plot(sc_logScale)
     ...: plt.plot(sc_moving_Average, color='blue')
     ...: #print(sc_rolmean,sc_rolstd)
Out[108]: [<matplotlib.lines.Line2D at 0x2288f1885f8>]
```

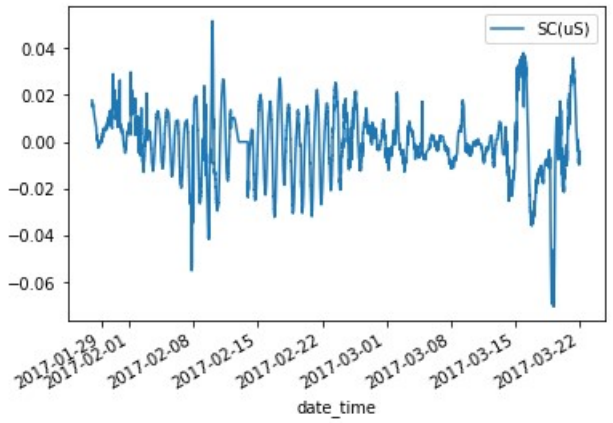



```
In [109]: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(12)
Out[109]:
                     SC(uS)
date_time
2017-01-27 00:00:00     NaN
2017-01-27 00:15:00     NaN
2017-01-27 00:30:00     NaN
2017-01-27 00:45:00     NaN
2017-01-27 01:00:00     NaN
2017-01-27 01:15:00     NaN
2017-01-27 01:30:00     NaN
2017-01-27 01:45:00     NaN
2017-01-27 02:00:00     NaN
2017-01-27 02:15:00     NaN
2017-01-27 02:30:00     NaN
2017-01-27 02:45:00     NaN

In [110]: sc_LogScaleMinusMovingAverage.head(100)
Out[110]:
                      SC(uS)
date_time
2017-01-27 00:00:00       NaN
2017-01-27 00:15:00       NaN
2017-01-27 00:30:00       NaN
2017-01-27 00:45:00       NaN
2017-01-27 01:00:00       NaN
2017-01-27 01:15:00       NaN
2017-01-27 01:30:00       NaN
2017-01-27 01:45:00       NaN
2017-01-27 02:00:00       NaN
2017-01-27 02:15:00       NaN
2017-01-27 02:30:00       NaN
2017-01-27 02:45:00       NaN
2017-01-27 03:00:00       NaN
2017-01-27 03:15:00       NaN
2017-01-27 03:30:00       NaN
2017-01-27 03:45:00       NaN
2017-01-27 04:00:00       NaN
2017-01-27 04:15:00       NaN
2017-01-27 04:30:00       NaN
2017-01-27 04:45:00       NaN
```

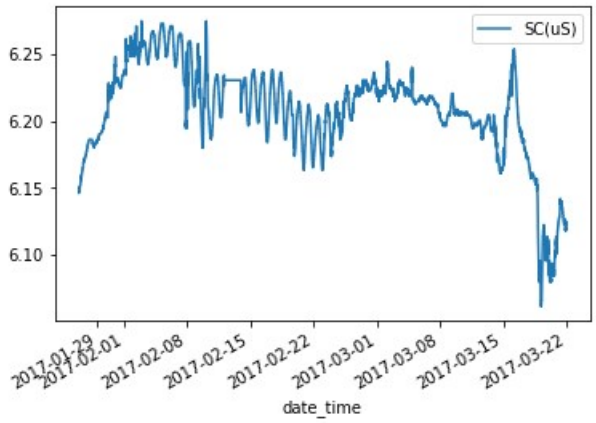
```
2017-01-27 05:00:00       NaN
2017-01-27 05:15:00       NaN
2017-01-27 05:30:00       NaN
2017-01-27 05:45:00       NaN
2017-01-27 06:00:00       NaN
2017-01-27 06:15:00       NaN
2017-01-27 06:30:00       NaN
2017-01-27 06:45:00       NaN
2017-01-27 07:00:00       NaN
2017-01-27 07:15:00       NaN
...                       ...
2017-01-27 17:45:00       NaN
2017-01-27 18:00:00       NaN
2017-01-27 18:15:00       NaN
2017-01-27 18:30:00       NaN
2017-01-27 18:45:00       NaN
2017-01-27 19:00:00       NaN
2017-01-27 19:15:00       NaN
2017-01-27 19:30:00       NaN
2017-01-27 19:45:00       NaN
2017-01-27 20:00:00       NaN
2017-01-27 20:15:00       NaN
2017-01-27 20:30:00       NaN
2017-01-27 20:45:00       NaN
2017-01-27 21:00:00       NaN
2017-01-27 21:15:00       NaN
2017-01-27 21:30:00       NaN
2017-01-27 21:45:00       NaN
2017-01-27 22:00:00       NaN
2017-01-27 22:15:00       NaN
2017-01-27 22:30:00       NaN
2017-01-27 22:45:00       NaN
2017-01-27 23:00:00       NaN
2017-01-27 23:15:00       NaN
2017-01-27 23:30:00       NaN
2017-01-27 23:45:00       NaN
2017-01-28 00:00:00  0.017694
2017-01-28 00:15:00  0.017300
2017-01-28 00:30:00  0.016906
2017-01-28 00:45:00  0.016512
2017-01-28 01:00:00  0.016118

[100 rows x 1 columns]

In [111]: sc_LogScaleMinusMovingAverage.dropna(inplace=true)
Traceback (most recent call last):

  File "<ipython-input-111-696c7b715384>", line 1, in <module>
    sc_LogScaleMinusMovingAverage.dropna(inplace=true)

NameError: name 'true' is not defined


In [112]:
```

```
In [112]: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
     ...:

In [113]: sc_LogScaleMinusMovingAverage.head(100)
     ...:
Out[113]:
                      SC(uS)
date_time
2017-01-28 00:00:00  0.017694
2017-01-28 00:15:00  0.017300
2017-01-28 00:30:00  0.016906
2017-01-28 00:45:00  0.016512
2017-01-28 01:00:00  0.016118
2017-01-28 01:15:00  0.015746
2017-01-28 01:30:00  0.015375
2017-01-28 01:45:00  0.015025
2017-01-28 02:00:00  0.014653
2017-01-28 02:15:00  0.016342
2017-01-28 02:30:00  0.015971
2017-01-28 02:45:00  0.015601
2017-01-28 03:00:00  0.015230
2017-01-28 03:15:00  0.014881
2017-01-28 03:30:00  0.014532
2017-01-28 03:45:00  0.014184
2017-01-28 04:00:00  0.013835
2017-01-28 04:15:00  0.013486
2017-01-28 04:30:00  0.013160
2017-01-28 04:45:00  0.012833
2017-01-28 05:00:00  0.012506
2017-01-28 05:15:00  0.012180
2017-01-28 05:30:00  0.011875
2017-01-28 05:45:00  0.011571
2017-01-28 06:00:00  0.011266
2017-01-28 06:15:00  0.010984
2017-01-28 06:30:00  0.010680
2017-01-28 06:45:00  0.010397
2017-01-28 07:00:00  0.010115
2017-01-28 07:15:00  0.009832
...                       ...
2017-01-28 17:30:00 -0.000941
2017-01-28 17:45:00 -0.001027
2017-01-28 18:00:00 -0.001113
2017-01-28 18:15:00 -0.001200
2017-01-28 18:30:00 -0.001265
2017-01-28 18:45:00 -0.001329
2017-01-28 19:00:00 -0.001394
2017-01-28 19:15:00 -0.001459
2017-01-28 19:30:00 -0.001524
2017-01-28 19:45:00 -0.001567
2017-01-28 20:00:00  0.000432
2017-01-28 20:15:00  0.000368
2017-01-28 20:30:00  0.000303
2017-01-28 20:45:00  0.000239
2017-01-28 21:00:00  0.000174
2017-01-28 21:15:00  0.000109
```

```
2017-01-28 21:30:00  0.000066
2017-01-28 21:45:00  0.000023
2017-01-28 22:00:00 -0.000020
2017-01-28 22:15:00 -0.000063
2017-01-28 22:30:00 -0.000084
2017-01-28 22:45:00 -0.000106
2017-01-28 23:00:00 -0.000127
2017-01-28 23:15:00 -0.000149
2017-01-28 23:30:00 -0.000170
2017-01-28 23:45:00 -0.000192
2017-01-29 00:00:00 -0.000192
2017-01-29 00:15:00  0.001846
2017-01-29 00:30:00 -0.000213
2017-01-29 00:45:00  0.001825

[100 rows x 1 columns]

In [114]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_LogScaleMinusMovingAverage['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     dfoutput['Critical Value (%s)'%key] = value
     ...:
     ...: print(dfoutput)
Results of Dickey Fuller Test:
Test Statistic                -1.236223e+01
p-value                        5.513780e-23
#Lags Used                     2.200000e+01
Number of Observations Used    5.053000e+03
Critical Value (1%)           -3.431645e+00
Critical Value (5%)           -2.862112e+00
Critical Value (10%)          -2.567075e+00
dtype: float64

In [115]: sc_LogScaleMinusMovingAverage.plot()
Out[115]: <matplotlib.axes._subplots.AxesSubplot at 0x2288442a860>
```




```
In [116]: sc_logScale.plot()
Out[116]: <matplotlib.axes._subplots.AxesSubplot at 0x22884ad5860>
```

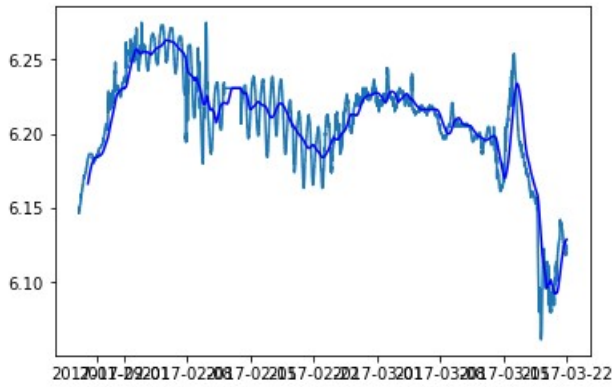



```
In [117]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_logScale['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     dfoutput['Critical Value (%s)'%key] = value
     ...:
     ...: print(dfoutput)
Results of Dickey Fuller Test:
Test Statistic                  -3.204103
p-value                          0.019748
#Lags Used                      27.000000
Number of Observations Used    5143.000000
Critical Value (1%)              -3.431622
Critical Value (5%)              -2.862102
Critical Value (10%)             -2.567069
dtype: float64

In [118]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     dfoutput['Critical Value (%s)'%key] = value
     ...:
     ...: print(dfoutput)
Results of Dickey Fuller Test:
Test Statistic                  -3.268401
p-value                          0.016351
#Lags Used                      27.000000
Number of Observations Used    5143.000000
Critical Value (1%)              -3.431622
Critical Value (5%)              -2.862102
Critical Value (10%)             -2.567069
dtype: float64

In [119]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_moving_Average['SC(uS)'], autolag='AIC')
```

```
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     dfoutput['Critical Value (%s)'%key] = value
     ...:
     ...: print(dfoutput)
Results of Dickey Fuller Test:
Traceback (most recent call last):

  File "<ipython-input-119-ee31616eccfe>", line 2, in <module>
    dftest = adfuller(sc_moving_Average['SC(uS)'], autolag='AIC')

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\tsa\stattools.py", line
241, in adfuller
    maxlag, autolag)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\tsa\stattools.py", line 86,
in _autolag
    mod_instance = mod(endog, exog[:, :lag], *modargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\regression
\linear_model.py", line 817, in __init__
    hasconst=hasconst, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\regression
\linear_model.py", line 663, in __init__
    weights=weights, hasconst=hasconst, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\regression
\linear_model.py", line 179, in __init__
    super(RegressionModel, self).__init__(endog, exog, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\model.py", line 212,
in __init__
    super(LikelihoodModel, self).__init__(endog, exog, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\model.py", line 64, in
__init__
    **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\model.py", line 87, in
_handle_data
    data = handle_data(endog, exog, missing, hasconst, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\data.py", line 633, in
handle_data
    **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\data.py", line 79, in
__init__
    self._handle_constant(hasconst)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\data.py", line 133, in
_handle_constant
```

```
    raise MissingDataError('exog contains inf or nans')

MissingDataError: exog contains inf or nans


In [120]:

In [120]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_movingSTD['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     dfoutput['Critical Value (%s)'%key] = value
     ...:
     ...: print(dfoutput)
Results of Dickey Fuller Test:
Traceback (most recent call last):

  File "<ipython-input-120-7bcf29101514>", line 2, in <module>
    dftest = adfuller(sc_movingSTD['SC(uS)'], autolag='AIC')

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\tsa\stattools.py", line
241, in adfuller
    maxlag, autolag)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\tsa\stattools.py", line 86,
in _autolag
    mod_instance = mod(endog, exog[:, :lag], *modargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\regression
\linear_model.py", line 817, in __init__
    hasconst=hasconst, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\regression
\linear_model.py", line 663, in __init__
    weights=weights, hasconst=hasconst, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\regression
\linear_model.py", line 179, in __init__
    super(RegressionModel, self).__init__(endog, exog, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\model.py", line 212,
in __init__
    super(LikelihoodModel, self).__init__(endog, exog, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\model.py", line 64, in
__init__
    **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\model.py", line 87, in
_handle_data
    data = handle_data(endog, exog, missing, hasconst, **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\data.py", line 633, in
```

```
handle_data
    **kwargs)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\data.py", line 79, in
__init__
    self._handle_constant(hasconst)

  File "C:\Users\admin\Anaconda3\lib\site-packages\statsmodels\base\data.py", line 133, in
_handle_constant
    raise MissingDataError('exog contains inf or nans')

MissingDataError: exog contains inf or nans


In [121]:

In [121]: sc_moving_Average = sc_logScale.rolling(window=95).mean() #window size 12
denotes 12 months,
     ...:

In [122]: sc_moving_Average = sc_logScale.rolling(window=95).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=95).std()
     ...: plt.plot(sc_logScale)
     ...: plt.plot(sc_moving_Average, color='blue')
Out[122]: [<matplotlib.lines.Line2D at 0x2288f5560f0>]
```




```
In [123]: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
Out[123]:
                     SC(uS)
date_time
2017-01-27 00:00:00      NaN
2017-01-27 00:15:00      NaN
2017-01-27 00:30:00      NaN
2017-01-27 00:45:00      NaN
2017-01-27 01:00:00      NaN
2017-01-27 01:15:00      NaN
2017-01-27 01:30:00      NaN
2017-01-27 01:45:00      NaN
2017-01-27 02:00:00      NaN
2017-01-27 02:15:00      NaN
```

```
2017-01-27 02:30:00       NaN
2017-01-27 02:45:00       NaN
2017-01-27 03:00:00       NaN
2017-01-27 03:15:00       NaN
2017-01-27 03:30:00       NaN
2017-01-27 03:45:00       NaN
2017-01-27 04:00:00       NaN
2017-01-27 04:15:00       NaN
2017-01-27 04:30:00       NaN
2017-01-27 04:45:00       NaN
2017-01-27 05:00:00       NaN
2017-01-27 05:15:00       NaN
2017-01-27 05:30:00       NaN
2017-01-27 05:45:00       NaN
2017-01-27 06:00:00       NaN
2017-01-27 06:15:00       NaN
2017-01-27 06:30:00       NaN
2017-01-27 06:45:00       NaN
2017-01-27 07:00:00       NaN
2017-01-27 07:15:00       NaN
...                       ...
2017-01-27 17:45:00       NaN
2017-01-27 18:00:00       NaN
2017-01-27 18:15:00       NaN
2017-01-27 18:30:00       NaN
2017-01-27 18:45:00       NaN
2017-01-27 19:00:00       NaN
2017-01-27 19:15:00       NaN
2017-01-27 19:30:00       NaN
2017-01-27 19:45:00       NaN
2017-01-27 20:00:00       NaN
2017-01-27 20:15:00       NaN
2017-01-27 20:30:00       NaN
2017-01-27 20:45:00       NaN
2017-01-27 21:00:00       NaN
2017-01-27 21:15:00       NaN
2017-01-27 21:30:00       NaN
2017-01-27 21:45:00       NaN
2017-01-27 22:00:00       NaN
2017-01-27 22:15:00       NaN
2017-01-27 22:30:00       NaN
2017-01-27 22:45:00       NaN
2017-01-27 23:00:00       NaN
2017-01-27 23:15:00       NaN
2017-01-27 23:30:00       NaN
2017-01-27 23:45:00  0.015816
2017-01-28 00:00:00  0.017482
2017-01-28 00:15:00  0.017084
2017-01-28 00:30:00  0.016686
2017-01-28 00:45:00  0.016288
2017-01-28 01:00:00  0.015912

[100 rows x 1 columns]
```

```
In [124]: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
```

```
In [125]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_LogScaleMinusMovingAverage['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     dfoutput['Critical Value (%s)'%key] = value
     ...:
     ...: print(dfoutput)
Results of Dickey Fuller Test:
Test Statistic                -1.247695e+01
p-value                        3.152817e-23
#Lags Used                     2.200000e+01
Number of Observations Used    5.054000e+03
Critical Value (1%)           -3.431645e+00
Critical Value (5%)           -2.862112e+00
Critical Value (10%)          -2.567075e+00
dtype: float64

In [126]:
```