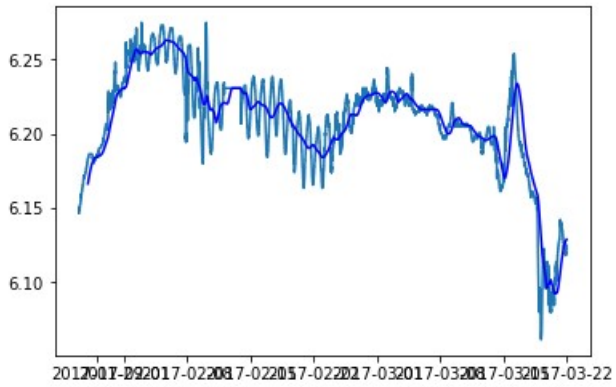
```
In [117]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_logScale['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     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
```

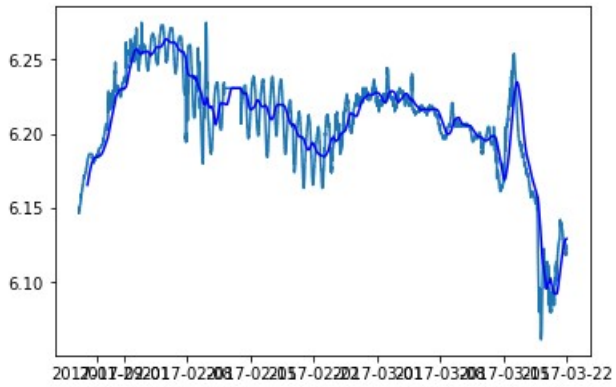
```
In [121]:

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
```

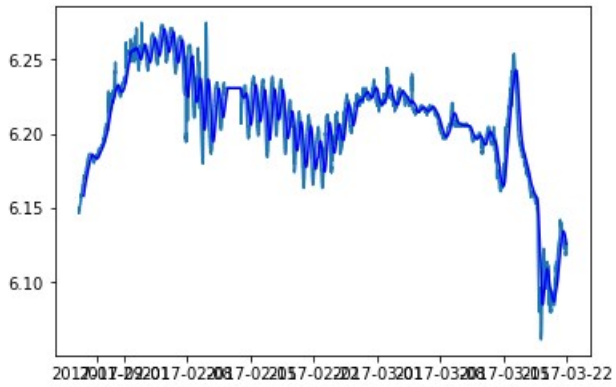
```
2017-01-27 06:00:00       NaN
2017-01-27 06:15:00       NaN
2017-01-27 06:30:00       NaN
2017-01-27 06:45:00       NaN
2017-01-27 07:00:00       NaN
2017-01-27 07:15:00       NaN
...                       ...
2017-01-27 17:45:00       NaN
2017-01-27 18:00:00       NaN
2017-01-27 18:15:00       NaN
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]: sc_moving_Average = sc_logScale.rolling(window=90).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=90).std()
     ...: plt.plot(sc_logScale)
     ...: plt.plot(sc_moving_Average, color='blue')
Out[126]: [<matplotlib.lines.Line2D at 0x2288f4f3128>]
```



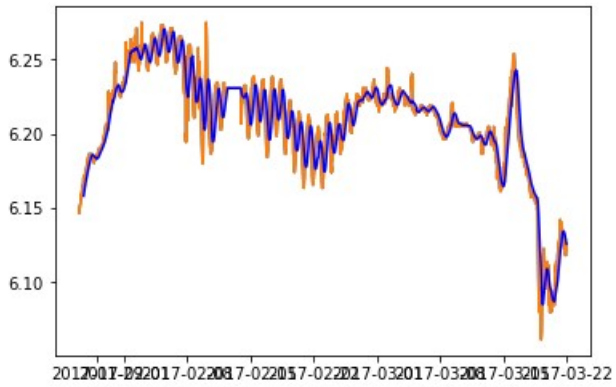

```
In [127]: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
     ...:
     ...: sc_LogScaleMinusMovingAverage.dropna(inplace=True)

In [128]: 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                -1.307040e+01
p-value                        1.962545e-24
#Lags Used                     2.200000e+01
Number of Observations Used    5.059000e+03
Critical Value (1%)           -3.431643e+00
Critical Value (5%)           -2.862111e+00
Critical Value (10%)          -2.567074e+00
dtype: float64

In [129]: sc_moving_Average = sc_logScale.rolling(window=50).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=90).std()
     ...: plt.plot(sc_logScale)
```

```
     ...: plt.plot(sc_moving_Average, color='blue')
     ...:
     ...: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
     ...:
     ...: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
```



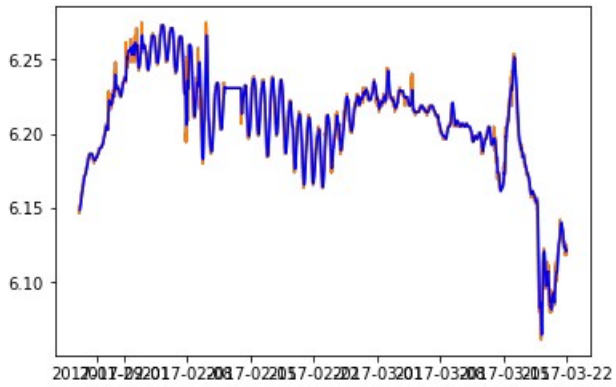

```
In [130]: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_LogScaleMinusMovingAverage['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
Used','Number of Observations Used'])
     ...: for key,value in dftest[4].items():
     ...:     dfoutput['Critical Value (%s)'%key] = value
     ...:
     ...: print(dfoutput)
Results of Dickey Fuller Test:
Test Statistic                -1.570493e+01
p-value                        1.381652e-28
#Lags Used                     3.100000e+01
Number of Observations Used    5.090000e+03
Critical Value (1%)           -3.431635e+00
Critical Value (5%)           -2.862108e+00
Critical Value (10%)          -2.567072e+00
dtype: float64

In [131]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
     ...: #Determine rolling statistics
     ...: sc_moving_Average = sc_logScale.rolling(window=50).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=50).std()
     ...: plt.plot(sc_logScale)
     ...: plt.plot(sc_moving_Average, color='blue')
     ...:
     ...: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
     ...:
     ...: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
     ...: #print(sc_rolmean,sc_rolstd)
     ...: 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                -1.570493e+01
p-value                        1.381652e-28
#Lags Used                     3.100000e+01
Number of Observations Used    5.090000e+03
Critical Value (1%)           -3.431635e+00
Critical Value (5%)           -2.862108e+00
Critical Value (10%)          -2.567072e+00
dtype: float64
```



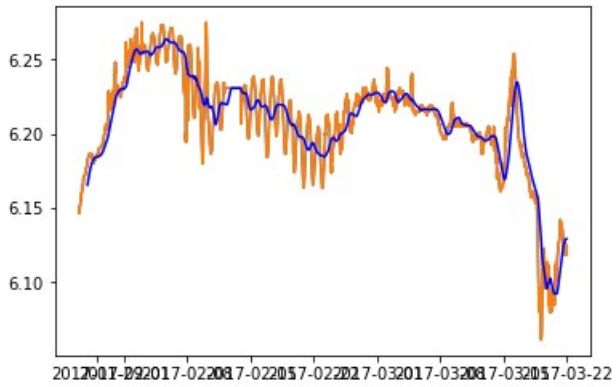

```
In [132]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
     ...: #Determine rolling statistics
     ...: sc_moving_Average = sc_logScale.rolling(window=12).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=12).std()
     ...: plt.plot(sc_logScale)
     ...: plt.plot(sc_moving_Average, color='blue')
     ...:
     ...: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
     ...:
     ...: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
     ...: #print(sc_rolmean,sc_rolstd)
     ...: 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                -1.502636e+01
p-value                        1.005071e-27
#Lags Used                     3.300000e+01
Number of Observations Used    5.126000e+03
Critical Value (1%)           -3.431626e+00
Critical Value (5%)           -2.862104e+00
Critical Value (10%)          -2.567070e+00
dtype: float64
```



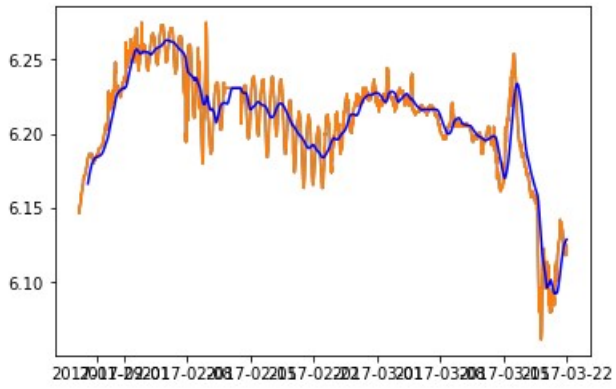

```
In [133]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
     ...: #Determine rolling statistics
     ...: sc_moving_Average = sc_logScale.rolling(window=90).mean() #window size 12
denotes 12 months,
     ...: #giving rolling mean at yearly level
     ...: sc_movingSTD = sc_logScale.rolling(window=90).std()
     ...: plt.plot(sc_logScale)
     ...: plt.plot(sc_moving_Average, color='blue')
     ...:
     ...: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
     ...:
     ...: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
     ...: #print(sc_rolmean,sc_rolstd)
     ...: 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                -1.307040e+01
p-value                        1.962545e-24
#Lags Used                     2.200000e+01
Number of Observations Used    5.059000e+03
```

```
Critical Value (1%)           -3.431643e+00
Critical Value (5%)           -2.862111e+00
Critical Value (10%)          -2.567074e+00
dtype: float64
```



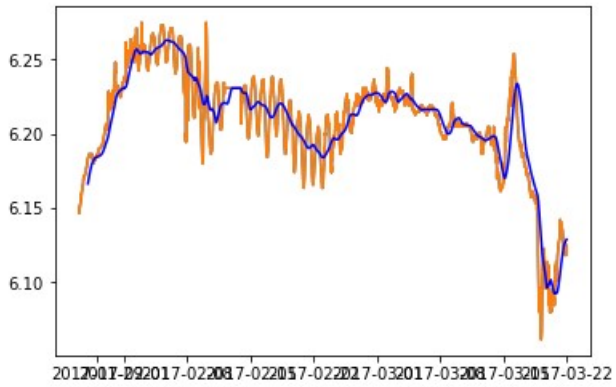

```
In [134]: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
     ...: #print(sc_rolmean,sc_rolstd)
     ...: print('Results of Dickey Fuller Test:')
     ...: dftest = adfuller(sc_logScale['SC(uS)'], autolag='AIC')
     ...:
     ...: dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags
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 [135]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
     ...: #Determine rolling statistics
     ...: 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')
     ...:
     ...: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
     ...:
     ...: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
     ...: #print(sc_rolmean,sc_rolstd)
```

```
     ...: 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 [136]: sc_logScale = np.log(sc)
     ...: plt.plot(sc_logScale)
     ...: #Determine rolling statistics
     ...: 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')
     ...:
     ...: sc_LogScaleMinusMovingAverage = sc_logScale - sc_moving_Average
     ...: sc_LogScaleMinusMovingAverage.head(100)
     ...:
     ...: sc_LogScaleMinusMovingAverage.dropna(inplace=True)
     ...: #print(sc_rolmean,sc_rolstd)
     ...: 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 [**137**]: