Time Series Analysis- GroupH - Assignment 2

Group H: Shenyi Mao, Yueting Jiang, Chenyu Zhao, Jose Ferreira

Question 1

```
In [1]: %matplotlib inline

        import numpy as np
        import matplotlib.pyplot as plt
        from statsmodels.tsa.ar_model import AR
        from statsmodels.tsa.arima_model import ARMA
        import statsmodels as sm
        import pandas as pd
```

```
In [2]: #Simulate AR(1)
        alpha=0.1
        beta = 0.3
        sigma=0.005
        T_list=[100,250,1250]
        N = 2000


        for T in T_list:
            alpha_sum = 0
            beta_sum = 0
            sigma_sum = 0
            for i in range(0,N):
                x0=alpha/(1-beta)
                x=np.zeros(T+1)
                x[0]=x0
                eps=np.random.normal(0.0,sigma,T)
                for i in range(1,T+1):
                    x[i]=alpha+beta*x[i-1]+eps[i-1]

                y=x[0:T]
                yp=x[1:(T+1)]
                m=np.sum(y)/T
                mp=np.sum(yp)/T
                betaCMLE=np.inner(y-m,yp-mp)/np.inner(y-m,y-m)
                alphaCMLE=mp-betaCMLE*m
                sigmaCMLE=np.sqrt(np.inner(yp-betaCMLE*y-alphaCMLE,
                                           yp-betaCMLE*y-alphaCMLE)/T)

                alpha_sum += alphaCMLE
                beta_sum += betaCMLE
                sigma_sum += sigmaCMLE
```

```
            alpha_hat = alpha_sum / N
            beta_hat = beta_sum / N
            sigma_hat = sigma_sum / N
            print("T = ", T, ",(alpha,beta,sigma) = " ,[alpha_hat,beta_hat,sigma_hat])
```

```
T =  100 ,(alpha,beta,sigma) =  [0.1024153805831, 0.2831247621365, 0.004951490497068]
T =  250 ,(alpha,beta,sigma) =  [0.1009745294900, 0.2932032633954, 0.004971390214618]
T =  1250 ,(alpha,beta,sigma) =  [0.1002329906964, 0.2983269719304, 0.004993956810083]
```

- $ Comment$
- We run simulations N(=2000) times to give T(=100,250,1250) observations from the given AR(1) model and then fit the observartions into AR(1) to give the average MLE estimates.
- The output is:

$$T = 100, [\hat{\alpha}, \hat{\beta}, \hat{\sigma}] = [0.102415, 0.283124, 0.00495149]$$

$$T = 250, [\hat{\alpha}, \hat{\beta}, \hat{\sigma}] = [0.100974, 0.293203, 0.00497139]$$

$$T = 1250, [\hat{\alpha}, \hat{\beta}, \hat{\sigma}] = [0.100232, 0.298326, 0.00499396]$$

  we can see that the estimates are biased compare with the given parameters

$$[\alpha, \beta, \sigma] = [0.1, 0.3, 0.005]$$

- for these biased estimates,we can see that when T goes larger, the bias get smaller because the error is O($\frac{1}{N}$),which means that they are consistant.

Question 2

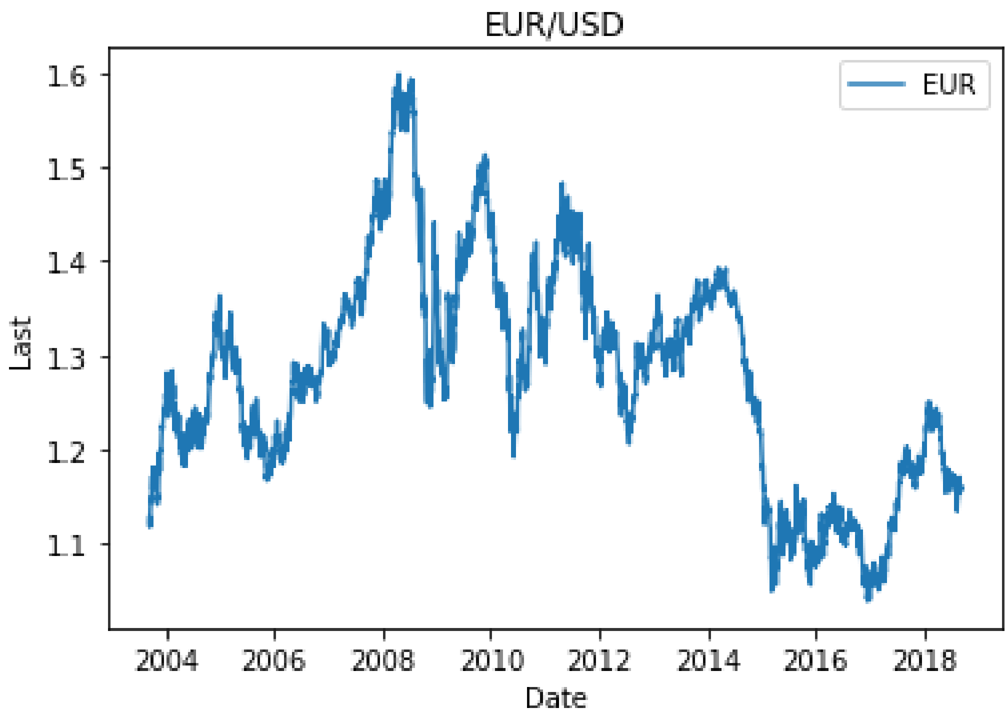
In [3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
```

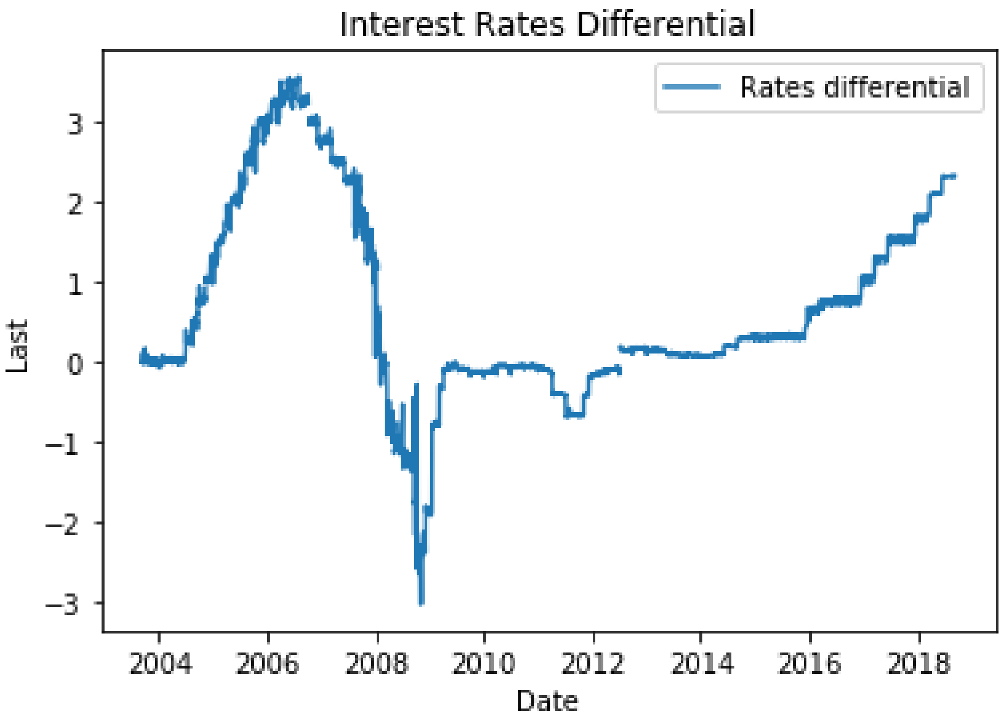
In [4]:
```
def perform_adf_test(xt):
    """
    input:  a timeseries representing the daily difference between SPY and IWV returns
    output: outputs the statistic values from the ADF test
    """
    test = adfuller(xt)
    print('ADF Statistic: %f' % test[0])
    print('p-value: %f' % test[1])
    for key, value in test[4].items():
        print('\t%s: %.3f' % (key, value))
```

```python
In [5]: def problem_2():
            # Section 2.1: Generate time series
            eur = pd.read_excel('HW2_Data.xlsx', sheet_name = 'EUR',
                                index_col=0, parse_dates=True, infer_datetime_format=True)
            fed = pd.read_excel('HW2_Data.xlsx', sheet_name = 'FEDL01',
                                index_col=0, parse_dates=True, infer_datetime_format=True)
            ecb = pd.read_excel('HW2_Data.xlsx', sheet_name = 'EUORDEPOT',
                                index_col=0, parse_dates=True, infer_datetime_format=True)

            diff = fed - ecb

            plt.figure()
            plt.plot(eur,label='EUR')
            plt.title('EUR/USD')
            plt.xlabel('Date')
            plt.ylabel('Last')
            plt.legend()
            plt.show()

            plt.plot(diff,label='Rates differential')
            plt.title('Interest Rates Differential')
            plt.xlabel('Date')
            plt.ylabel('Last')
            plt.legend()
            plt.show()

            # run Dickey-Fuller on each series separately.
            eur_rates = eur['Last Price'].dropna()
            diff_interest_rates = diff['Last Price'].dropna()
            print("EURUSD")
            perform_adf_test(eur_rates)
            print("Diff")
            perform_adf_test(diff_interest_rates)

            # this should be stationary
            print("delta EURUSD")
            lagged_eur = eur['Last Price'].pct_change(1).dropna()
            perform_adf_test(lagged_eur)

            # this too should be stationary
            print("delta rates")
            lagged_rates = diff['Last Price'].pct_change(1).replace([np.inf, -np.inf], np.nan).d
            perform_adf_test(lagged_rates)

            # form vector a = (1, -1)' ==> u_t = x - y
            merged_data = pd.concat([eur_rates.rename('eur'), diff_interest_rates.rename('rates'
            clean_data = merged_data.dropna()
            u = (clean_data['eur'] - clean_data['rates'])
```

```
    # test unit root on u, this would have to be stationary
    print("u")
    perform_adf_test(u)
```

```
In [9]: problem_2()
```






```
EURUSD
ADF Statistic: -1.966646
p-value: 0.301386
        5%: -2.862
        1%: -3.432
        10%: -2.567
Diff
ADF Statistic: -1.016293
p-value: 0.747276
        5%: -2.862
        1%: -3.432
        10%: -2.567
delta EURUSD
ADF Statistic: -62.486528
p-value: 0.000000
        5%: -2.862
        1%: -3.432
        10%: -2.567
delta rates
ADF Statistic: -9.344125
p-value: 0.000000
        5%: -2.862
        1%: -3.432
        10%: -2.567
```

```
u
ADF Statistic: -0.930077
p-value: 0.777906
        5%: -2.862
        1%: -3.432
        10%: -2.567
```

Comment

First, we run ADF on EURUUSD, the short interest difference between Fed and ECB. The test result shows that both series are not stationary at 5% significance level. And the first difference of both series are stationary at 5% significance level. Thus we can conclude that both series are I(1).

Second, we use the given cointegration vector $\alpha = (1, -1)$ to calculate the residual of the cointegration process u_t, which $u_t = EURUSD - Diff$. Then we test the stationarity of the reriduals. The test shows that the residual is not stationary at 5% significance level. Thus the vector $\alpha = (1, -1)$ is not the cointegration vector of EURUSD exchange rate and the differential between Fed and ECB short interest rates in these currencies.

TS. HW2. Group H.

Q3: i we first calculate $p(x_1, x_2 \cdots x_T \mid \theta, x_0)$

$$p(x_1 \cdots x_T \mid x_0, \theta) = \frac{1}{(2\pi\sigma^2)^{T/2}} \cdot \exp\left(-\frac{1}{2\sigma^2}\sum_{t=1}^{T} \varepsilon_t^2\right).$$

(Note that $\hat{\varepsilon}_t = x_t - \alpha - \beta x_{t-1} - \delta t \sim N(0, \sigma^2)$).

Thus. $p(x_1 \cdots x_T \mid x_0, \theta) = \frac{1}{(2\pi\sigma^2)^{T/2}} \cdot \exp\left(-\frac{1}{2\sigma^2}\sum_{t=1}^{T}(x_t - \alpha - \beta x_{t-1} - \delta t)^2\right).$

ii draw log-likelihood estimation.

$$-\log L(\theta \mid y) = \frac{1}{2} T \log \sigma^2 + \frac{1}{2\sigma^2}\sum_{t=1}^{T}(x_t - \alpha - \beta x_{t-1} - \delta t)^2 + \text{constant}$$

Minimizing this function yields:

$$\begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \\ \hat{\delta} \end{pmatrix} = \begin{pmatrix} T & \sum_{t=1}^{T} x_{t-1} & \sum_{t=1}^{T} t \\ \sum_{t=1}^{T} x_{t-1} & \sum_{t=1}^{T} x_{t-1}^2 & \sum_{t=1}^{T} t x_{t-1} \\ \sum_{t=1}^{T} t & \sum_{t=1}^{T} t x_{t-1} & \sum_{t=1}^{T} t^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_{t=1}^{T} x_t \\ \sum_{t=1}^{T} x_t x_{t-1} \\ \sum_{t=1}^{T} t \cdot x_t \end{pmatrix}.$$

$$\hat{\sigma}^2 = \frac{1}{T} \sum_{t=1}^{T}(x_t - \hat{\alpha} - \hat{\beta} x_{t-1} - \hat{\delta} t)^2.$$