2018.12.18-week_02-collaborative_review_assignment-neelkanth_metha

December 17, 2018

1 Validation, Interpretation, Implication!

Validation, interpretation, and implication are three of the most important steps in analyzing any statistical or economic model. Having introduced the Aggregate Supply-Aggregate Demand Model in the previous notes and the impacts of Cost-Push Inflation, the major question is whether this model serves as an accurate description of the economy and its response to changes in oil prices and Monetarist Policy. In order to evaluate the AS-AD Model, we are going to simulate its response to changes in oil prices and broad money supply using real economic data and compare it to data on real GDP and price-level in order to graphical test for the model's validity. Based on our historical discussions earlier in this module, we will be comparing Reserve Bank responses to economic stagflation brought on by increasing oil prices. In order to identify the effects of Monetarist Policy, we will look at the different approaches taken by Reserve Banks in the United States and the United Kingdom. Much of the data used in this analysis will be sourced from the World Bank, World Development Indicators, available through the pandas_datareader API. Due to changes in the API and its dependencies, students may need to read through the documentation it requires. A known issue is this API's compatibility issue with the newer version of the Pandas library. A comment has been added on this in the code and a specific change to one of the Pandas Modules has been added to aid in this compatibility. You will see this across the notes.

In some cases, data has been scaled in order to aid in interpretability, as we are concerned with the effect and not the specific value for broad money or oil prices. Where possible these changes have been flagged and are crucial for their inclusion in later AS-AD Model. Our libraries remain the same as the previous set of notes, using NumPy, SciPy, HoloViews, and Pandas, with much of the code repeated from before.

```
In [2]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import pandas_datareader.wb as wb
        import holoviews as hv
        import hvplot.pandas
        from scipy.optimize import fsolve
        from scipy.stats import iqr
        %matplotlib inline
```

```
In [3]: # There is a compatibility issue with this library \
        #and newer versions of Pandas, this is short fix to the problem, \
        #if you have issues at this chunk comment it out and you should be fine.
        pd.core.common.is_list_like = pd.api.types.is_list_like
        # import pandas_datareader.wb as wb

In [4]: hv.extension('bokeh')
        np.random.seed(42)

In [5]: def P(*args, **kwargs):
            P = np.linspace(-10, 10, 100).reshape(-1,1)
            P = P[P!=0]
            return P

        def AS(P=P(), W=0, P_e=1, Z_2=0):
            return P-Z_2

        def AD(P=P(), M=0, G=0, T=0, Z_1=0):
            return -P+Z_1

In [6]: def findIntersection(fun1,fun2,x0):
            return fsolve(lambda x : fun1(x) - fun2(x),x0)

In [7]: indicators = wb.get_indicators()
        indicators.head()

Out[7]:                       id                                    name  \
        0    1.0.HCount.1.90usd          Poverty Headcount ($1.90 a day)
        1     1.0.HCount.2.5usd          Poverty Headcount ($2.50 a day)
        2  1.0.HCount.Mid10to50    Middle Class ($10-50 a day) Headcount
        3       1.0.HCount.Ofcl  Official Moderate Poverty Rate-National
        4    1.0.HCount.Poor4uds             Poverty Headcount ($4 a day)

                    source                                       sourceNote  \
        0  LAC Equity Lab  The poverty headcount index measures the propo...
        1  LAC Equity Lab  The poverty headcount index measures the propo...
        2  LAC Equity Lab  The poverty headcount index measures the propo...
        3  LAC Equity Lab  The poverty headcount index measures the propo...
        4  LAC Equity Lab  The poverty headcount index measures the propo...

                                     sourceOrganization    topics unit
        0  b'LAC Equity Lab tabulations of SEDLAC (CEDLAS...  Poverty
        1  b'LAC Equity Lab tabulations of SEDLAC (CEDLAS...  Poverty
        2  b'LAC Equity Lab tabulations of SEDLAC (CEDLAS...  Poverty
        3  b'LAC Equity Lab tabulations of data from Nati...  Poverty
        4  b'LAC Equity Lab tabulations of SEDLAC (CEDLAS...  Poverty

In [8]: indicators.loc[indicators.id =='NY.GDP.PETR.RT.ZS',:]

Out[8]:                       id                name                        source  \
        7886  NY.GDP.PETR.RT.ZS  Oil rents (% of GDP)  World Development Indicators
```

```
                                                  sourceNote  \
7886  Oil rents are the difference between the value...

                                          sourceOrganization  \
7886  b'World Bank staff estimates based on sources ...

                                    topics unit
7886  Energy & Mining  ; Environment
```

```
In [9]: countries = wb.get_countries()
        countries.head()
```

```
Out[9]:                                   adminregion capitalCity iso3c  \
        0                                               Oranjestad   ABW
        1                                  South Asia       Kabul   AFG
        2                                                           AFR
        3    Sub-Saharan Africa (excluding high income)      Luanda   AGO
        4  Europe & Central Asia (excluding high income)      Tirane   ALB

                   incomeLevel iso2c  latitude     lendingType  longitude  \
        0          High income    AW  12.51670  Not classified   -70.0167
        1           Low income    AF  34.52280             IDA    69.1761
        2           Aggregates    A9       NaN      Aggregates        NaN
        3  Lower middle income    AO  -8.81155            IBRD    13.2420
        4  Upper middle income    AL  41.33170            IBRD    19.8172

                  name                      region
        0        Aruba  Latin America & Caribbean
        1  Afghanistan                 South Asia
        2       Africa                 Aggregates
        3       Angola          Sub-Saharan Africa
        4      Albania       Europe & Central Asia
```

The World Bank data-portal offers a wide array of data points used to benchmark various interventions, economic growth, prosperity, education, and healthcare. This data is extensive but often inconsistent across countries or years. For this reason, it some cases, it is required to find appropriate proxies to capture Macroeconomic variables discussed in academia. For tracking the real cost of oil over time, we will use the reserve banks measure of 'oil rents (% of GDP)' which is defined as "... the difference between the value of crude oil production at regional prices and total costs of production". Data was sourced between 1970 to 2017 for use in the analysis in order to get a long run understanding of the events which led up to the crash of 1987 and the implication monetary policy and commodity prices may have on markets today.

The graph below shows an accurate measure for the changing economic cost of oil experienced by particular countries over time. The graph details clear spikes in oil rents in both the US and UK, starting at around 1976, at the time of the first recession. These reach their peak for either country at around 1980 and 1985 for the US and UK, before restoring to long-run norms. Zooming and panning through the data, you can observe greater detail in these movements as they changed over time and better understand the events at the time of the crash.

```
In [10]: %%opts Curve [width=800, height=450]
         oil = wb.download(
             indicator='NY.GDP.PETR.RT.ZS',
             country=['USA','GBR'],
             start=pd.to_datetime('1970', yearfirst=True),
             end=pd.to_datetime('2017', yearfirst=True)
         )
         oil = oil.reset_index().dropna()

         oil_unscaled = oil

         oil.loc[oil.country=='United States', 'NY.GDP.PETR.RT.ZS'] = (oil.loc[oil.country=='Uni
                                                                       oil.loc[oil.country=='Un
                                                                        iqr(oil.loc[oil.country

         oil.loc[oil.country=='United Kingdom', 'NY.GDP.PETR.RT.ZS'] = (oil.loc[oil.country=='Un
                                                                       oil.loc[oil.country=='Un
                                                                        iqr(oil.loc[oil.country

         oil_plot = oil.iloc[::-1,:].hvplot.line(x='year', y='NY.GDP.PETR.RT.ZS', by='country',

         oil_plot

Out[10]: :NdOverlay   [country]
            :Curve   [year]   (NY.GDP.PETR.RT.ZS)
```

In order to track Reserve Bank policy, we need to observe its effect on money markets and on money supply. For use in these notes, we will be looking at M4 or broad money, defined as "the sum of currency outside banks; demand deposits other than those of the central government; the time, savings, and foreign currency deposits of resident sectors other than the central government; bank and traveler's checks; and other securities such as certificates of deposit and commercial paper". While for specific Reserve Bank interventions M0 and M1 money may be more appropriate, to understanding its implication M4 money will be used to understand its effects through the money multiplier effect. From the graphs below, we can see around the time of 1985 strongly diverging changes in broad money between these two economic. While the UK sees stabilizing and increasing broad money, the US experiences a drastic decline in broad money as a percent of GDP, which continued well into the 1990s.

```
In [11]: %%opts Curve [width=800, height=450]
         money = wb.download(indicator='FM.LBL.BMNY.GD.ZS', country=['USA','GBR'], start=pd.to_d
         money = money.reset_index().dropna()

         money_unscaled = money

         money.loc[money.country=='United States', 'FM.LBL.BMNY.GD.ZS'] = (money.loc[money.count
                                                                           money.loc[money.count
                                                                             iqr(money.loc[money
        money.loc[money.country=='United Kingdom', 'FM.LBL.BMNY.GD.ZS'] = (money.loc[money.coun
                                                                             money.loc[money.count
                                                                               iqr(money.loc[money

        money_plot = money.iloc[::-1,:].hvplot.line(x='year', y='FM.LBL.BMNY.GD.ZS', by='countr

        money_plot
```

```
Out[11]: :NdOverlay   [country]
            :Curve   [year]   (FM.LBL.BMNY.GD.ZS)
```

We will be comparing this variable against real GDP or real output using a constant 2010 USD Price-Level. Given the relationship between real GDP and our AS-AD model, we will prefer to look at real GDP per capita growth due to its scaling and interpretability. From the graphs below, it is clear that a strong correlation exists between these two countries across time, with clear evidence of recessions in 1987 and 1974.

```
In [12]: %%opts Curve [width=800, height=450]
         gdp = wb.download(indicator='NY.GDP.PCAP.KD', country=['USA','GBR'], start=pd.to_dateti
         gdp = gdp.reset_index()

         gdp.loc[:,'NY.GDP.PCAP.KD'] = gdp.loc[:,'NY.GDP.PCAP.KD'].pct_change()

         gdp = gdp.loc[pd.to_numeric(gdp.year)<=2012,:].dropna()

         gdp_plot = gdp.iloc[::-1,:].hvplot.line(x='year', y='NY.GDP.PCAP.KD', by='country', tit

         gdp_plot
```

```
Out[12]: :NdOverlay   [country]
            :Curve   [year]   (NY.GDP.PCAP.KD)
```

In the interactive plot below we will use scaled values for broad money and oil rents as values of Z_2 and Z_1 for use in our AS-AD model. Using the slider, we can more these exogenous shocks through time, observing their effect on Price-level and real GDP output. Using the real output set at equilibria between this price we scale this equilibrium and compare it against our real GDP, shown by the red dot in the right panel of the graph, to analyze the validity of this model for use across a range of applications. We compare these models for the UK and the US to arrive at some conclusion around the effects of Monetary Policy on the real economy and capital markets. These models do not take into account all variable but aim to approximate an estimate of these models' predictions.

```
In [13]: def curves_data_US(year=1971):

             oil_z2 = oil.loc[oil.country=='United States', 'NY.GDP.PETR.RT.ZS'].iloc[::-1]
             oil_z2 = oil_z2 - oil_z2.iloc[0]

             money_z2 = money.loc[money.country=='United States', 'FM.LBL.BMNY.GD.ZS'].iloc[::-1
             money_z2 = money_z2 -money_z2.iloc[0]
```

```
            z_2 = oil_z2.iloc[year-1971] -10

            z_1= -money_z2.iloc[year-1971]-10

            as_eq = pd.DataFrame([P(), AS(P=P(), Z_2=0)], index=['Price-Level','Real Output']).
            ad_eq = pd.DataFrame([P(), AD(P=P(), Z_1=0)], index=['Price-Level','Real Output']).

            as_shock = pd.DataFrame([P(), AS(P=P(), Z_2=z_2+10)], index=['Price-Level','Real Ou
            ad_shock = pd.DataFrame([P(), AD(P=P(), Z_1=z_1+10)], index=['Price-Level','Real Ou

            result = findIntersection(lambda x: AS(P=x, Z_2=z_2+10), lambda x: AD(P=x, Z_1=-z_1
            r = result + 1e-4 if result==0 else result

            plot = hv.Curve(as_eq, vdims='Price-Level',kdims='Real Output').options(alpha=0.2,
                                 hv.Curve(ad_eq, vdims='Price-Level',kdims='Real Output').
                                 hv.Curve(as_shock, vdims='Price-Level',kdims='Real Output
                                 hv.Curve(ad_shock, vdims='Price-Level',kdims='Real Output
                                 hv.VLine(-result[0]).options(color='black', alpha=0.2, li
                                 hv.HLine(AD(P=-r[0], Z_1=z_1+10)).options(color='black',

            gdp_mean = gdp.loc[gdp.country=='United States', 'NY.GDP.PCAP.KD'].iloc[0]
            gdp_iqr = iqr(gdp.loc[gdp.country=='United States', 'NY.GDP.PCAP.KD'])

            gdp_plot_US = gdp.loc[gdp.country=='United States',:].iloc[::-1,:].hvplot.line(x='y
            hv.VLine(year).options(color='black') * pd.DataFrame([[(AD(P=-r[0], Z_1=-z_1-10))*g

            return plot.options(xticks=[0], yticks=[0], title_format="US Short-Run AS-AD Model"
```

```
In [14]: %%opts Curve [width=400, height=400]

         hv.DynamicMap(curves_data_US, kdims=['year'], label="US Short-Run AS-AD Model")\
         .redim.range(year=(1971,2007))
```

```
Out[14]: :DynamicMap   [year]
            :Layout
               .Overlay.I  :Overlay
                  .Curve.I  :Curve   [Real Output]   (Price-Level)
                  .Curve.II :Curve   [Real Output]   (Price-Level)
                  .Curve.AS :Curve   [Real Output]   (Price-Level)
                  .Curve.AD :Curve   [Real Output]   (Price-Level)
                  .VLine.I  :VLine   [x,y]
                  .HLine.I  :HLine   [x,y]
               .Overlay.II :Overlay
                  .Curve.I    :Curve   [year]   (NY.GDP.PCAP.KD)
                  .VLine.I    :VLine   [x,y]
                  .Scatter.I :Scatter   [year]   (Real Output)
```

Similarly, for the US, the AS-AD model appears fairly accurate in tracking overall trends in the data, despite certicular failures at points in time. Despite it simplicity, the AS-AD model is able to capture the important Macroeconomic dymamics providing predictive insight into the effects of global politics and macroeconomic policy on a countries economy. It is clear that while countries respond differently to Monetary intervention, such interventions play a crucial role in managing economic crisis which we will explore in greater depth in later modules.

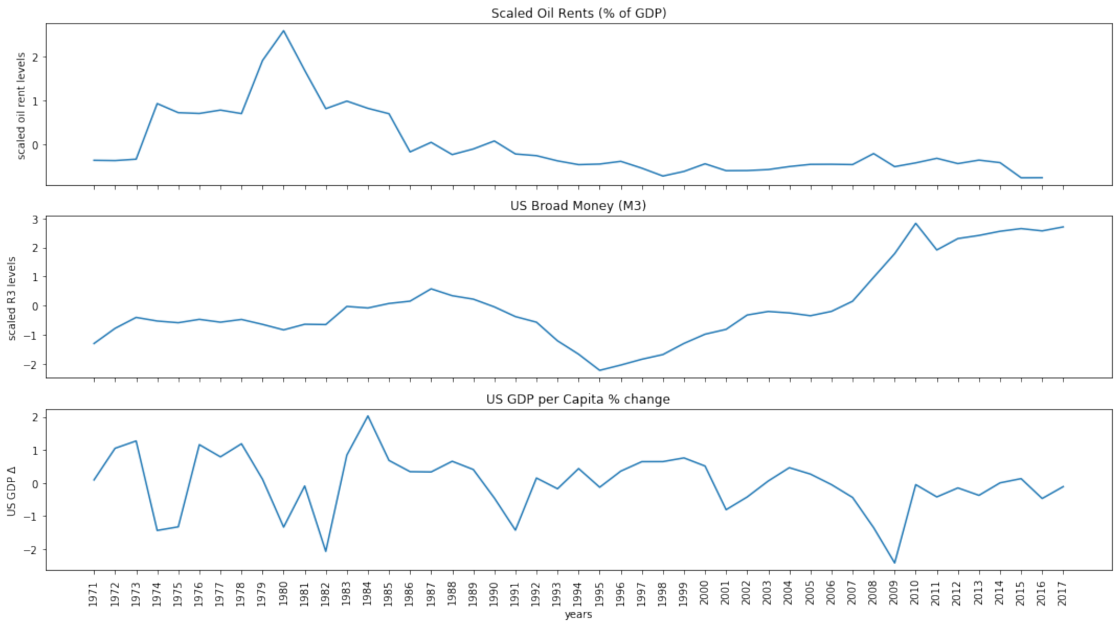
Fitting model

```
In [15]: '''US Oil Rents (% of GDP)'''
         # Importing Oil rents for US
         USoil = wb.download(indicator='NY.GDP.PETR.RT.ZS', country='USA', start=pd.to_datetime(

         # Cleaning the downloaded dataset
         USoil = (
             USoil[::-1]
             .reset_index(level=0, drop=True)
             .rename(columns={'NY.GDP.PETR.RT.ZS':'USoil'})
             .dropna(axis=0)
         )

         # obtaining IQR data for US oil rents
         USoilIQR = (USoil - USoil.mean())/ iqr(x=USoil)

         '''M3 broad money supply'''
         # Importing US broad money (M3)
         USmoney = wb.download(indicator='FM.LBL.BMNY.GD.ZS', country='USA', start=pd.to_datetim

         # Cleaning the downloaded dataset
         USmoney = (
             USmoney[::-1]
             .reset_index(level=0, drop=True)
             .rename(columns={'FM.LBL.BMNY.GD.ZS':'US M3'})
             .dropna(axis=0)
         )

         # Obtaining IQR for US M3
         USmoneyIQR = (USmoney - USmoney.mean())/ iqr(x=USmoney)

         '''US GDP per Capita'''
         # Importing US GDP per capita growth
         USGDP = wb.download(indicator='NY.GDP.PCAP.KD', country='USA', start=pd.to_datetime('19

         # Cleaning the downloaded dataset
         USGDP = (
             USGDP[::-1]
             .reset_index(level=0, drop=True)
             .rename(columns={'NY.GDP.PCAP.KD':'US_GDP'})
             .dropna(axis=0)
```

```
            .pct_change()[1:]
        )

        # Obtaining IQR for US M3
        USGDPIQR = (USGDP - USGDP.mean())/ iqr(x=USGDP)
```

Observing all the variables independently

```
In [16]: fig, axes = plt.subplots(nrows=3, ncols=1, sharex=True, figsize=(15,8))
         axes[0].plot(USoilIQR, label='US oil rents')
         axes[0].set_title('Scaled Oil Rents (% of GDP)')
         axes[0].set_ylabel('scaled oil rent levels')
         axes[1].plot(USmoneyIQR, label='US M3')
         axes[1].set_title('US Broad Money (M3)')
         axes[1].set_ylabel('scaled R3 levels')
         axes[2].plot(USGDPIQR, label='US GDP per Capita')
         axes[2].set_title('US GDP per Capita % change')
         axes[2].set_ylabel('US GDP $\Delta$')
         plt.tight_layout(True)
         plt.xticks(rotation=90)
         plt.xlabel('years')
         plt.show()
```

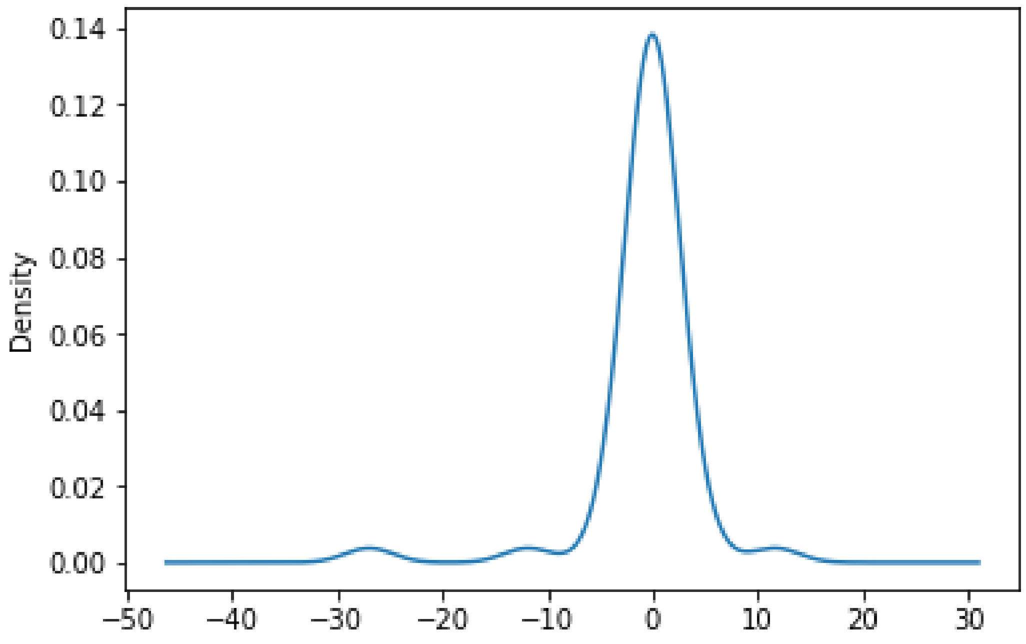


The GDP per capita appears stationary, but the same can't be concluded for both the exogenous variable. This violates gaussian and IID assumption in linear regression.

```
In [17]: df = pd.merge(left=USoilIQR, right=USmoneyIQR, left_index=True, right_index=True, how='
         X = df['USoil']/ df['US M3']
```

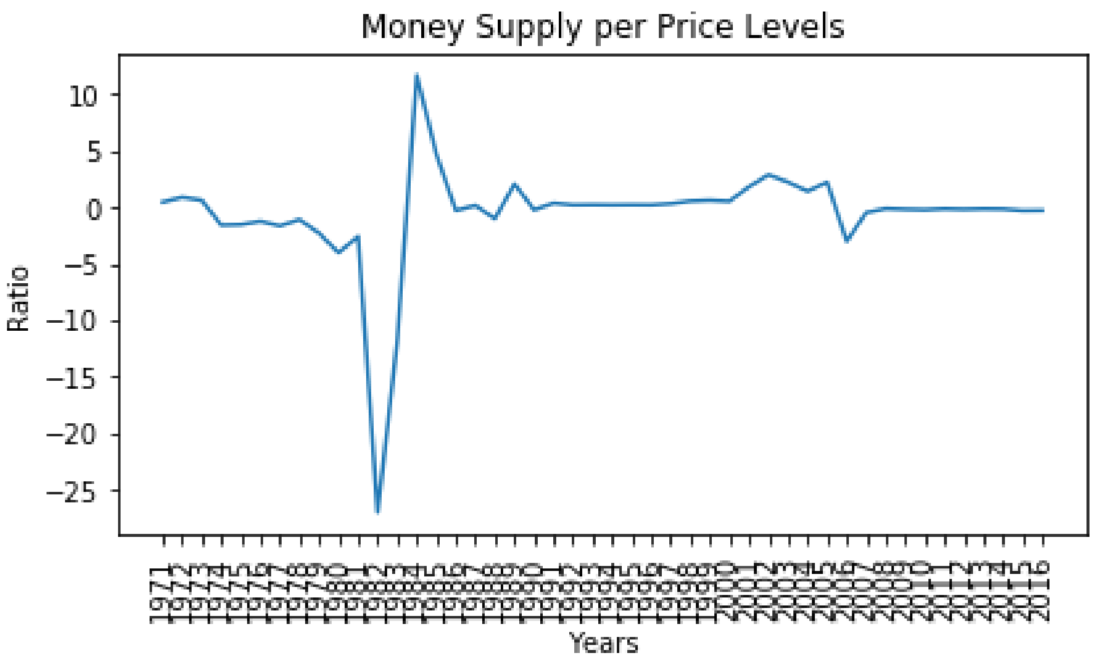
```
        y = USGDPIQR[:-1]

        # X = pd.merge(left=oil, right=money, left_index=True, right_index=True, how='inner')
        # y = GDP[:-1]
```

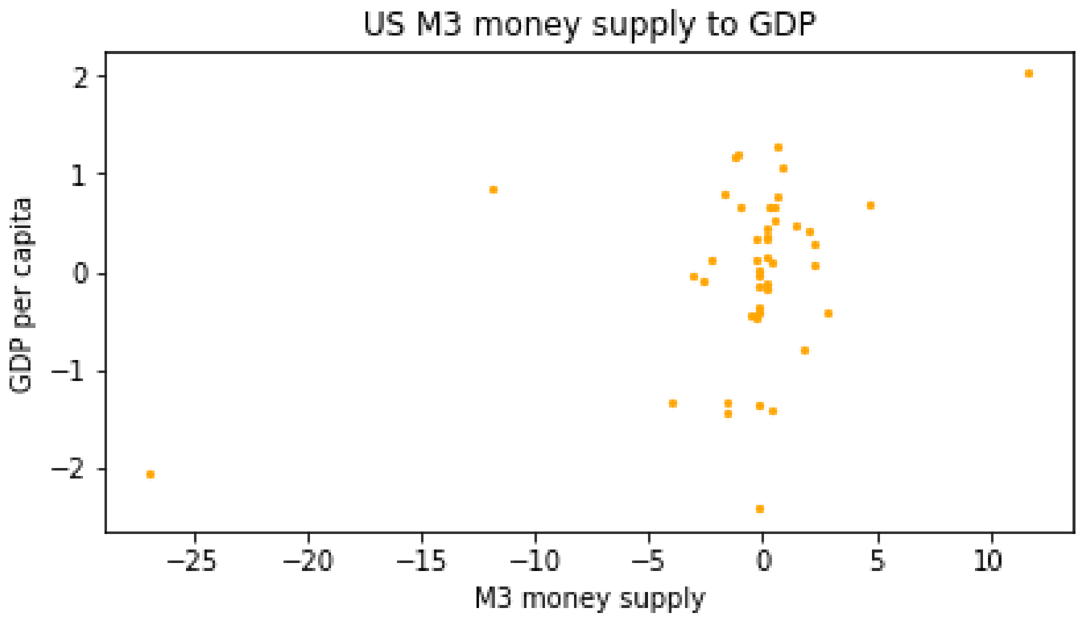
```
In [18]: X.plot.kde();
```



With X as the ratio of money supply to price levels, the series is nearly normalized. The regression result obtained would be a little more reliable.

```
In [19]: fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(6,7))
         plt.subplot(211)
         plt.plot(X, label='USoil/ US M3');
         plt.title('Money Supply per Price Levels')
         plt.xlabel('Years')
         plt.ylabel('Ratio')
         plt.xticks(rotation=90)
         plt.subplot(212)
         plt.scatter(x=X, y=y, s=5, color='orange');
         plt.title('US M3 money supply to GDP')
         plt.xlabel('M3 money supply')
         plt.ylabel('GDP per capita')
         plt.tight_layout(True)
         plt.show()
```





```
In [20]: from statsmodels.regression import linear_model
         from statsmodels.tools import eval_measures

In [21]: model = linear_model.OLS(endog=y, exog=X,hasconst=False)
         regr = model.fit()

         regr.summary2()
```

```
Out[21]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                          Results: Ordinary least squares
         =================================================================
         Model:              OLS              Adj. R-squared:     0.146
         Dependent Variable: US_GDP           AIC:                112.9201
         Date:               2018-12-17 16:28 BIC:                114.7488
         No. Observations:   46               Log-Likelihood:     -55.460
         Df Model:           1                F-statistic:        8.879
         Df Residuals:       45               Prob (F-statistic): 0.00464
         R-squared:          0.165            Scale:              0.66725
         --------------------------------------------------------------------
                   Coef.     Std.Err.      t       P>|t|     [0.025    0.975]
         --------------------------------------------------------------------
         x1        0.0732      0.0246    2.9798    0.0046    0.0237    0.1226
         -----------------------------------------------------------------
         Omnibus:               4.616        Durbin-Watson:           1.300
         Prob(Omnibus):         0.099        Jarque-Bera (JB):        3.411
         Skew:                  -0.584       Prob(JB):                0.182
         Kurtosis:              3.646        Condition No.:           1
         =================================================================

         """
```

```
In [22]: y_pred = pd.DataFrame(data=regr.predict(exog=X), columns=['y_pred'], index=y.index)
```

```
In [23]: print("RMSE: {}".format(eval_measures.rmse(x1=y, x2=y_pred)[0]))
```

```
RMSE: 0.8079257628947749
```

Comments The following could be inferred from the regression summary above: 1. The model params are:

$$G\hat{D}P_i = 0.0732 \times \frac{M3_i}{IQR(USoil)_i}$$

2. Price levels are negatively correlated to AggDem (GDP); i.e., other things being equal, at higher price levels lower the level of goods/ services are demanded and *vice versa*. (The oil rents % of GDP are assumed to be part of price levels though the oil supply shocks are the exogenous factor). 3. Money supply is positively correlated with GDP; i.e., for greater liquidity in the economy, the level of demand for goods/ services increases. 4. The exogenous variable in the above regression equation is the ratio of money supply to price levels. >1 implies favorable liquidity in the economy; conversly, <1 implies lower liquidity levels. 4. In ecnonmic parlence; GDP is fuction of the prevalent money supply per prevalent price levels, Govt. Spending, Govt. Income (real Tax), investment levels and other exogenous factors not captured above.

$$
\begin{aligned}
Y &= Y^d(\frac{M}{P}, G, T, I, Z_1) && (1)\\
\text{Where...} & && (2)\\
Y &: \text{real GDP} && (3)\\
\frac{M}{P} &: \text{the ratio of money supply and price levels} && (4)\\
G &: \text{Govt. Spending} && (5)\\
T &: \text{Real Tax component} && (6)\\
I &: \text{Real Investments} && (7)\\
Z_1 &: \text{exogenous factors not captured above} && (8)
\end{aligned}
$$

5. The StdErr is 0.0246, t-score 2.9798 and $P > |t|$ was recorded at 0.0046 - all indicating a robustness of the result.
6. The reason for low $\bar{R}$ is due to the fact that other variables in the AggDem are not considered in our analysis. (out of scope of the assignment). This means, the F-stat, although at comfortable level, has scope for improvement, with addition of other variables in the equation.
7. The skewness score and excess kurtosis, confirms my above claim that the data is close to normal. The effect of slightly non-normal skewness and kurtosis could be observed in > 0 Jarque-bera statistic.
8. The Durbin-Watson test-statistic of 1.3 indicates there is some level of serial autocorrelation present in the data. This is regerred to as stickyness. The reason for the stickyness in the data is due to the lag in effect of the policy implementation.

UK We'll now conduct similar analysis for UK data (and possibly if time permits for rest of the countries asked for in the assignment).

```
In [24]: def curves_data_UK(year=1971):

             oil_z2 = oil.loc[oil.country=='United Kingdom', 'NY.GDP.PETR.RT.ZS'].iloc[::-1]
             oil_z2 = oil_z2 - oil_z2.iloc[0]

             money_z2 = money.loc[money.country=='United Kingdom', 'FM.LBL.BMNY.GD.ZS'].iloc[::-
             money_z2 = money_z2 -money_z2.iloc[0]

             z_2 = oil_z2.iloc[year-1971] -10

             z_1= money_z2.iloc[year-1971]-10

             as_eq = pd.DataFrame([P(), AS(P=P(), Z_2=0)], index=['Price-Level','Real Output']).
             ad_eq = pd.DataFrame([P(), AD(P=P(), Z_1=0)], index=['Price-Level','Real Output']).

             as_shock = pd.DataFrame([P(), AS(P=P(), Z_2=z_2+10)], index=['Price-Level','Real Ou
             ad_shock = pd.DataFrame([P(), AD(P=P(), Z_1=z_1+10)], index=['Price-Level','Real Ou

             result = findIntersection(lambda x: AS(P=x, Z_2=z_2+10), lambda x: AD(P=x, Z_1=-z_1
    r = result + 1e-4 if result==0 else result

    plot = hv.Curve(as_eq, vdims='Price-Level',kdims='Real Output').options(alpha=0.2,
                             hv.Curve(ad_eq, vdims='Price-Level',kdims='Real Output').
                             hv.Curve(as_shock, vdims='Price-Level',kdims='Real Output
                             hv.Curve(ad_shock, vdims='Price-Level',kdims='Real Output
                             hv.VLine(-result[0]).options(color='black', alpha=0.2, li
                             hv.HLine(AS(P=-r[0], Z_2=-z_2-10)).options(color='black',

    gdp_mean = gdp.loc[gdp.country=='United Kingdom', 'NY.GDP.PCAP.KD'].iloc[0]
    gdp_iqr = iqr(gdp.loc[gdp.country=='United Kingdom', 'NY.GDP.PCAP.KD'])

    gdp_plot_UK = gdp.loc[gdp.country=='United Kingdom',:].iloc[::-1,:].hvplot.line(x='
    hv.VLine(year).options(color='black') * pd.DataFrame([[(AD(P=r[0], Z_1=z_1+10)*gdp_

    return plot.options(xticks=[0], yticks=[0], title_format="UK Short-Run AS-AD Model"
```

```
In [25]: %%opts Curve [width=400, height=400]

         hv.DynamicMap(curves_data_UK, kdims=['year'], label="UK Short-Run AS-AD Model")\
         .redim.range(year=(1971,2007))
```

```
Out[25]: :DynamicMap   [year]
            :Layout
               .Overlay.I  :Overlay
                  .Curve.I  :Curve   [Real Output]   (Price-Level)
                  .Curve.II :Curve   [Real Output]   (Price-Level)
                  .Curve.AS :Curve   [Real Output]   (Price-Level)
                  .Curve.AD :Curve   [Real Output]   (Price-Level)
                  .VLine.I  :VLine   [x,y]
                  .HLine.I  :HLine   [x,y]
               .Overlay.II :Overlay
                  .Curve.I   :Curve   [year]   (NY.GDP.PCAP.KD)
                  .VLine.I   :VLine   [x,y]
                  .Scatter.I :Scatter   [year]   (Real Output)
```

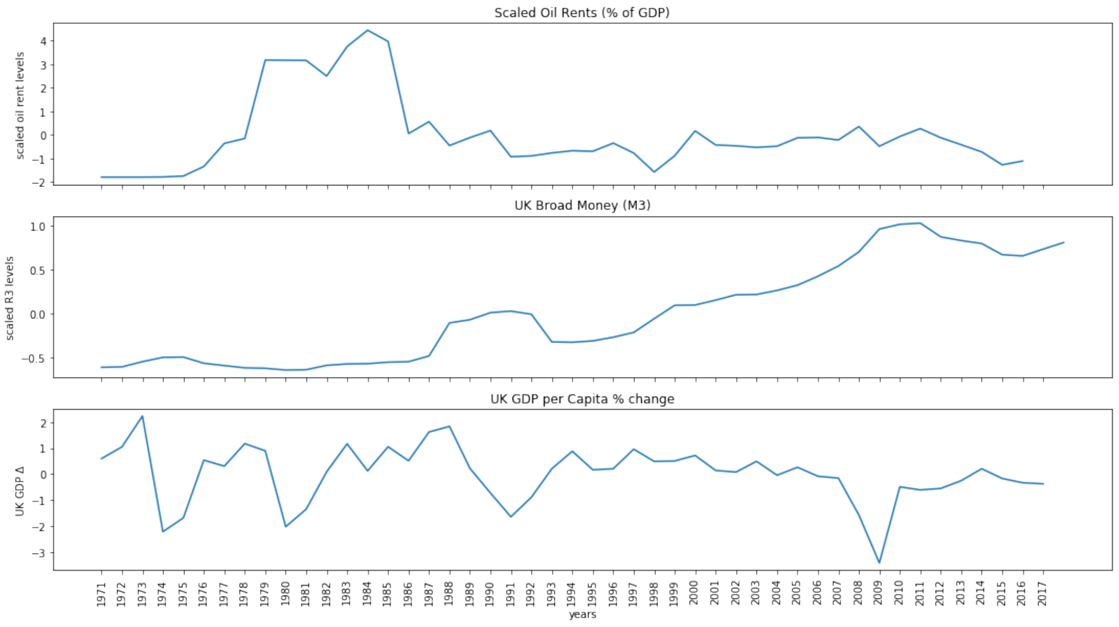
Looking at the graphs for UK GDP per capita growth and our AS-AD model, it is clear that while our model fails to account for the peaks in GDP per capita growth in 1973 and 1979, it does appear stationary at those points in time indicative of our comparison between real GDP and real GDP growth camputed in this graph. Overall the model seems to account well for the overall trend, including growth in 1990 in the mid-2000s.

```
In [26]: '''UK Oil Rents (% of GDP)'''
         # Importing Oil rents for US
         UKoil = wb.download(indicator='NY.GDP.PETR.RT.ZS', country='GBR', start=pd.to_datetime(

         # Cleaning the downloaded dataset
         UKoil = (
             UKoil[::-1]
```

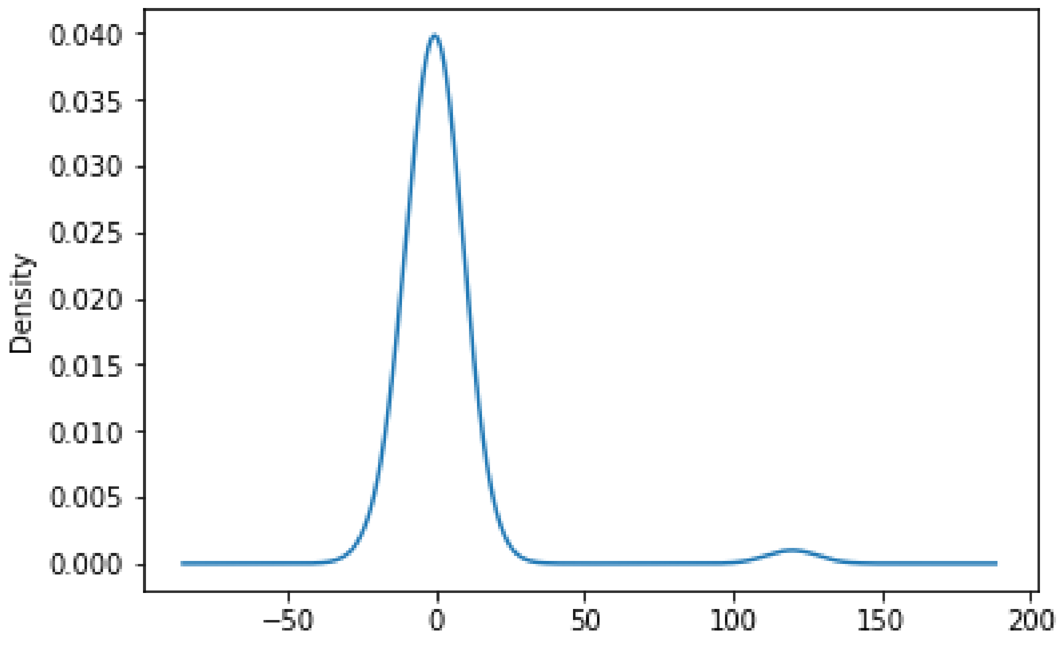
```
    .reset_index(level=0, drop=True)
    .rename(columns={'NY.GDP.PETR.RT.ZS':'UKoil'})
    .dropna(axis=0)
)

# obtaining IQR data for US oil rents
UKoilIQR = (UKoil - UKoil.mean())/ iqr(x=UKoil)

'''M3 broad money supply'''
# Importing US broad money (M3)
UKmoney = wb.download(indicator='FM.LBL.BMNY.GD.ZS', country='GBR', start=pd.to_datetim

# Cleaning the downloaded dataset
UKmoney = (
    UKmoney[::-1]
    .reset_index(level=0, drop=True)
    .rename(columns={'FM.LBL.BMNY.GD.ZS':'UK M3'})
    .dropna(axis=0)
)

# Obtaining IQR for US M3
UKmoneyIQR = (UKmoney - UKmoney.mean())/ iqr(x=UKmoney)

'''UK GDP per Capita'''
# Importing US GDP per capita growth
UKGDP = wb.download(indicator='NY.GDP.PCAP.KD', country='GBR', start=pd.to_datetime('19

# Cleaning the downloaded dataset
UKGDP = (
    UKGDP[::-1]
    .reset_index(level=0, drop=True)
    .rename(columns={'NY.GDP.PCAP.KD':'UK_GDP'})
    .dropna(axis=0)
    .pct_change()[1:]
)

# Obtaining IQR for US M3
UKGDPIQR = (UKGDP - UKGDP.mean())/ iqr(x=UKGDP)
```

In [27]:
```
fig, axes = plt.subplots(nrows=3, ncols=1, sharex=True, figsize=(15,8))
axes[0].plot(UKoilIQR, label='UK oil rents')
axes[0].set_title('Scaled Oil Rents (% of GDP)')
axes[0].set_ylabel('scaled oil rent levels')
axes[1].plot(UKmoneyIQR, label='UK M3')
axes[1].set_title('UK Broad Money (M3)')
axes[1].set_ylabel('scaled R3 levels')
axes[2].plot(UKGDPIQR, label='UK GDP per Capita')
axes[2].set_title('UK GDP per Capita % change')
```

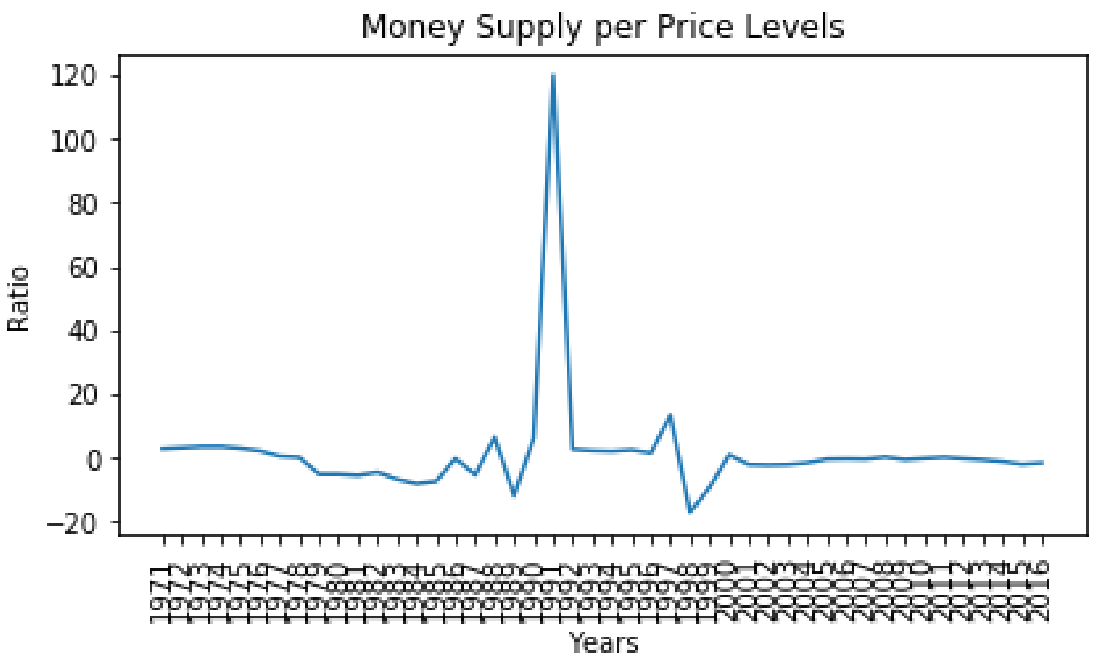
```
axes[2].set_ylabel('UK GDP $\Delta$')
plt.tight_layout(True)
plt.xticks(rotation=90)
plt.xlabel('years')
plt.show()
```

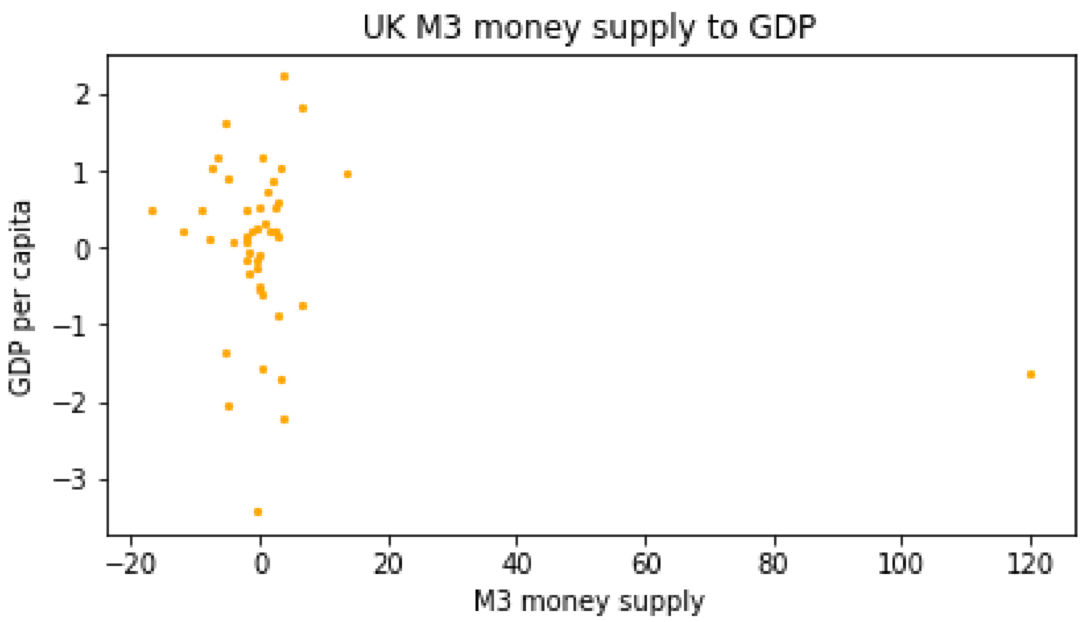


```
In [28]: df = pd.merge(left=UKoilIQR, right=UKmoneyIQR, left_index=True, right_index=True, how='
         X = df['UKoil']/ df['UK M3']
         y = UKGDPIQR[:-1]

         # X = pd.merge(left=oil, right=money, left_index=True, right_index=True, how='inner')
         # y = GDP[:-1]

In [29]: X.plot.kde();
```



```
In [30]: fig, axes = plt.subplots(nrows=2, ncols=1, figsize=(6,7))
         plt.subplot(211)
         plt.plot(X, label='UKoil/ UK M3');
         plt.title('Money Supply per Price Levels')
         plt.xlabel('Years')
         plt.ylabel('Ratio')
         plt.xticks(rotation=90)
         plt.subplot(212)
         plt.scatter(x=X, y=y, s=5, color='orange');
         plt.title('UK M3 money supply to GDP')
         plt.xlabel('M3 money supply')
         plt.ylabel('GDP per capita')
         plt.tight_layout(True)
         plt.show()
```





```
In [31]: model = linear_model.OLS(endog=y, exog=X,hasconst=False)
         regr = model.fit()

         regr.summary2()

Out[31]: <class 'statsmodels.iolib.summary2.Summary'>
         """
                             Results: Ordinary least squares
```

```
=================================================================
Model:              OLS              Adj. R-squared:     0.027
Dependent Variable: UK_GDP           AIC:                137.8405
Date:               2018-12-17 16:28 BIC:                139.6692
No. Observations:   46               Log-Likelihood:     -67.920
Df Model:           1                F-statistic:        2.282
Df Residuals:       45               Prob (F-statistic): 0.138
R-squared:          0.048            Scale:              1.1470
--------------------------------------------------------------------
        Coef.     Std.Err.       t       P>|t|      [0.025    0.975]
--------------------------------------------------------------------
x1     -0.0130     0.0086    -1.5107    0.1379    -0.0303    0.0043
-----------------------------------------------------------------
Omnibus:               9.795        Durbin-Watson:          1.312
Prob(Omnibus):         0.007        Jarque-Bera (JB):       9.753
Skew:                  -0.838       Prob(JB):               0.008
Kurtosis:              4.509        Condition No.:          1
=================================================================

"""
```

```
In [32]: y_pred = pd.DataFrame(data=regr.predict(exog=X), columns=['y_pred'], index=y.index)

In [33]: print("RMSE: {}".format(eval_measures.rmse(x1=y, x2=y_pred)[0]))

RMSE: 1.059279703581883
```

Comments The following could be inferred from the regression summary above: 1. The model params are:

$$UK\hat{G}DP_i = -0.0130 \times \frac{M3_i}{IQR(UKoil)_i}$$

2. The StdErr is 0.0086, t-score -1.5107 and $P > |t|$ was recorded at 0.1379 - all indicating a robustness of the result. 3. The $\bar{R}_i$ for the UK GDP model is too negligible for the model to be considered. There is no clear trend indicated by the AD-AS model. 4. There is greater skewness and excess kurtosis observed in the UK data as compared to US data. The Jarque-Bera test reflects this fact in $3x$ higher score compared to US data. 5. The Durbin-Watson test-statistic of ~1.3 indicates there is some level of serial autocorrelation present in the data. This is regerred to as stickyness. The reason for the stickyness in the data is due to the lag in effect of the policy implementation.

```
In [ ]:
```