Certificate in Data Analytics for Finance – UCD Professional Academy

Final Project

An Investigation to Determine the Efficacy of Natural Gas Prices as a Proxy When Forecasting Wholesale Electricity Prices

By Donal Smith

GitHub Repository: https://github.com/donalsmith/UCDPA_donal_smith.git

Table of Contents

1. Introduction

Electricity is an integral aspect of the world we live in, both on an individual level and a societal level. In the modern world, electricity is a basic human need and is considered one of the most important factors of societal progress. It's supply and availability plays a big role in shaping economic landscape and global opportunities for growth. In recent decades, electricity has entered the market as a tradable commodity and the power industries of many countries have experienced massive deregulation. This has paved the way for the modern day integrated competitive power markets that we see today, as well as the trade opportunities that result. Due to the competitive landscape of the power markets, the need for reliable forecasting methods at all scales (intra-day, daily, weekly, longer term) has also emerged and has been a large area of research.

The sources of UK electricity has in recent times shifted towards renewable generation with the uptake in generation technologies such as Solar, Wind and Biomass. Fossil fuel sources still account for 47% in 2018, Evans (2019). While coal generation dropped by 25%, NG dropped only 4%, accounting for 40% and maintain its identity as a primary generator. This is further supported by OxfordEnergy (2015), outlining the increase in reliance of NG for electricity generation from 1980 (0%) to 2015 (38%). A similar trend of NG sourcing for electricity generation is seen in Ireland, accounting for 42% of all generation in 2015 Kernan (2017). The significance of these proportions, would indicate some level of influence on the price of electricity generation.

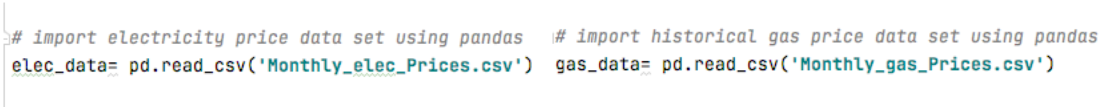
This brief paper will examine the efficacy of using Natural Gas (NG) prices as a proxy variable in the construction of Wholesale Electricity (WE) price forecasts. The hard analysis within this paper will be completed on the PyCharm platform using Python packages including Pandas, NumPy, Matplotlib, Statsmodels and more. After outlining the Data Sourcing process, this paper will then proceed in its examination in the following order: Data Exploration & Sourcing, Data Cleaning and Initial-Analysis, Visual Analysis and Data Plotting, Seasonality Analysis, Comparative Analysis and Correlation and Conclusion.

2. Data Exploration and Sourcing

This paper includes the use of two datasets – the first is a time series of historical UK Monthly Average Wholesale Electricity Prices, the second is a time series of historical UK Natural Gas monthly futures contracts prices, specifically the ICE UK Natural Gas contracts. Both datasets were exported from their sources as .csv files.

- 'Monthly_elec_Prices.csv'
- 'Monthly_gas_Prices.csv'.

The electricity prices dataset was exported from the market data resources of Nord Pool. Nord Pool is a European power exchange owned by Euronext and the continental Nordic and Baltic countries' Transmission system operators. Nord Pool delivers power trading across Europe and the UK. The (NG) prices dataset was exported from online markets data providers BarChart. Both datasets were imported using the pandas .csv command.



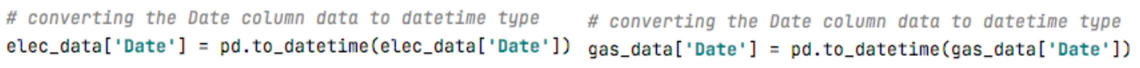
```
# import electricity price data set using pandas
elec_data= pd.read_csv('Monthly_elec_Prices.csv')
```

```
# import historical gas price data set using pandas
gas_data= pd.read_csv('Monthly_gas_Prices.csv')
```

3. Methodology

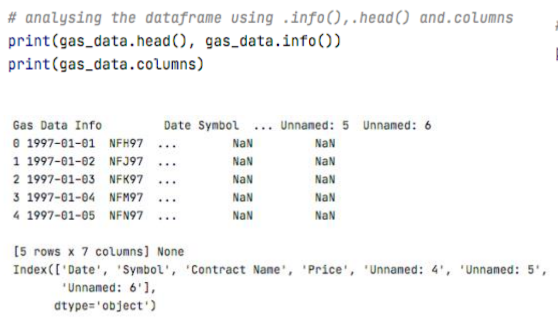
3.1 Data Cleaning and Initial Analysis

After importing both datasets using pandas the next step was to convert the date columns to datetime objects. This is achieved using the pd.to_datetime command. After this, a print command is called to retrieve some information about the time series'. The .head( ) command allows us to view the first five rows of the dataset which helps get a clear understanding of the data you intend to work with through visual representation, while the .info( ) command produces some additional information.



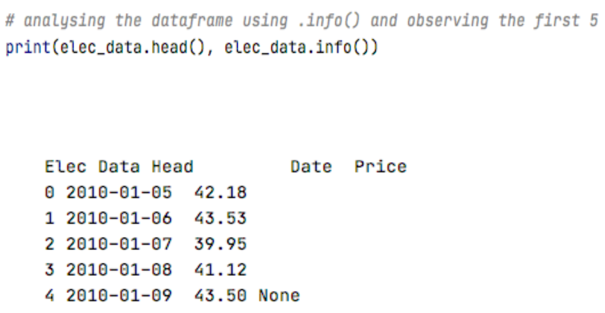
```
# converting the Date column data to datetime type
elec_data['Date'] = pd.to_datetime(elec_data['Date'])
```

```
# converting the Date column data to datetime type
gas_data['Date'] = pd.to_datetime(gas_data['Date'])
```



```
# analysing the dataframe using .info(),.head() and.columns
print(gas_data.head(), gas_data.info())
print(gas_data.columns)
```

```
Gas Data Info         Date Symbol  ... Unnamed: 5  Unnamed: 6
0 1997-01-01  NFH97  ...        NaN         NaN
1 1997-01-02  NFJ97  ...        NaN         NaN
2 1997-01-03  NFK97  ...        NaN         NaN
3 1997-01-04  NFM97  ...        NaN         NaN
4 1997-01-05  NFN97  ...        NaN         NaN

[5 rows x 7 columns] None
Index(['Date', 'Symbol', 'Contract Name', 'Price', 'Unnamed: 4', 'Unnamed: 5',
       'Unnamed: 6'],
      dtype='object')
```



```
# analysing the dataframe using .info() and observing the first 5
print(elec_data.head(), elec_data.info())
```

```
Elec Data Head          Date  Price
0 2010-01-05  42.18
1 2010-01-06  43.53
2 2010-01-07  39.95
3 2010-01-08  41.12
4 2010-01-09  43.50 None
```

We can see above that the NG df has seven columns total whereas the WE df has just two. The next step required in cleaning the data is to check for any missing values and to get a count. It is also important to check for duplicate values in the data and drop them if necessary (i.e. columns 5 – 7 of NG data).

```
# checking for missing values and finding the sum
print(elec_data.isnull().sum())
print(gas_data.isnull().sum())
```

```
# dropping missing values
gas_data = gas_data.dropna()
print(gas_data.isnull().sum())
```

```
Elec Null Date      0
Price     0
```

```
Gas Null Date            0
Symbol              0
Contract Name       0
Price               0
Unnamed: 4        287
Unnamed: 5        287
Unnamed: 6        287
```

As we only require the 'Date' and 'Price' column, we now must sort the NG df. This is done by creating a sorted df. In doing this, we must create a list of the required columns (Date and Price) in the call argument and then save the sorted df in csv format.

```
# sorting the gas data for columns
gas_sorted= gas_data[['Date','Price']]
print('Gas Sorted',gas_sorted.head())
```

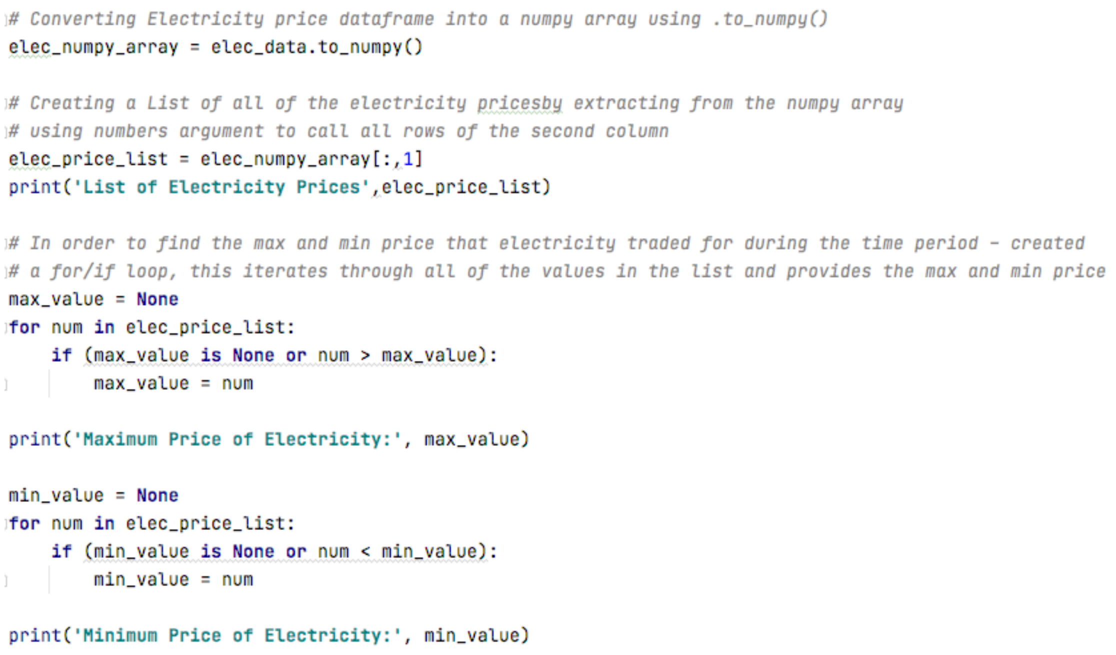
```
# Saving the sorted gas data in csv format
gas_data.to_csv("gasdata.csv")
```

```
Gas Sorted         Date    Price
0 1997-01-01  10.120
1 1997-01-02  10.000
2 1997-01-03  10.600
3 1997-01-04  10.625
4 1997-01-05   9.300
```

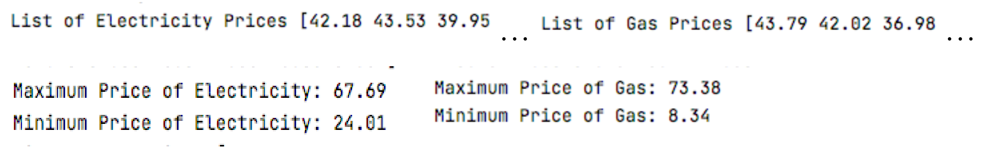
The final task before plotting was to examine the maximum and minimum prices for both NG and WE in their datasets. This allows us to get a clear understanding of the range of the data before plotting which is important in any analysis.

To do this we start by converting the price columns from the df's to NumPy arrays using .to_numpy( ). Converting to NumPy arrays allows us to extract a list for each, which includes all of their prices. This is done by retrieving a list of all rows (:) and the second column (1), which in this case is the 'Price' column. It is important to use (1) when calling for the second column as python is zero indexed. In order to find the maximum and minimum prices we then created two for-loops – one for the maximum function and one for the minimum. These for-loops then iterate through the respective lists of prices and provide us with the max/min price data.

The max value for-loop is structured so that it iterates through all the values in the list, if num is greater than the max_value, then that number becomes the new max value. Once the loop has iterated through all values in the list, it will print the maximum value that it encountered in the list that we created. The minimum value for-loop works in the same way but for less than values. The same process was repeated for both the WE df and the NG sorted df. The output results of the range analysis are detailed below. We can see electricity prices exhibited a high of £67.69/MWh and a low of £24.01/MWh, while NG prices exhibited a high of £73.38/MWh and a low of £8.34/MWh.



```python
# Converting Electricity price dataframe into a numpy array using .to_numpy()
elec_numpy_array = elec_data.to_numpy()

# Creating a List of all of the electricity pricesby extracting from the numpy array
# using numbers argument to call all rows of the second column
elec_price_list = elec_numpy_array[:,1]
print('List of Electricity Prices',elec_price_list)

# In order to find the max and min price that electricity traded for during the time period - created
# a for/if loop, this iterates through all of the values in the list and provides the max and min price
max_value = None
for num in elec_price_list:
    if (max_value is None or num > max_value):
        max_value = num

print('Maximum Price of Electricity:', max_value)

min_value = None
for num in elec_price_list:
    if (min_value is None or num < min_value):
        min_value = num

print('Minimum Price of Electricity:', min_value)
```



```
List of Electricity Prices [42.18 43.53 39.95 ...        List of Gas Prices [43.79 42.02 36.98 ...

Maximum Price of Electricity: 67.69      Maximum Price of Gas: 73.38
Minimum Price of Electricity: 24.01      Minimum Price of Gas: 8.34
```

3.2 Visual Analysis and Data Plotting

After the initial data cleaning and manipulation, both df's are now ready for visual analysis. When we plot data we can conduct a visual analysis and this allows us gain a better understanding of the data and to quickly identify any potential relationships, trends or patterns. Often times we are looking to identify price trends, volume trends and market or industry trends when using visual analysis. To create the plot we used the matplotlib package.

```
# plot data using matplotlib setting titles and color
elec_data.plot('Date','Price', title ='Monthly Electricity Prices (£/MW)', color = 'b')
plt.xlabel('Time (Months)')
plt.ylabel('Price (£)')
plt.show()

# plot data using matplotlib setting titles and color
gas_data.plot('Date','Price', title ='Monthly Gas Prices (£/MW)', color = 'orange')
plt.xlabel('Time (Months)')
plt.ylabel('Price (£)')
plt.show()
```

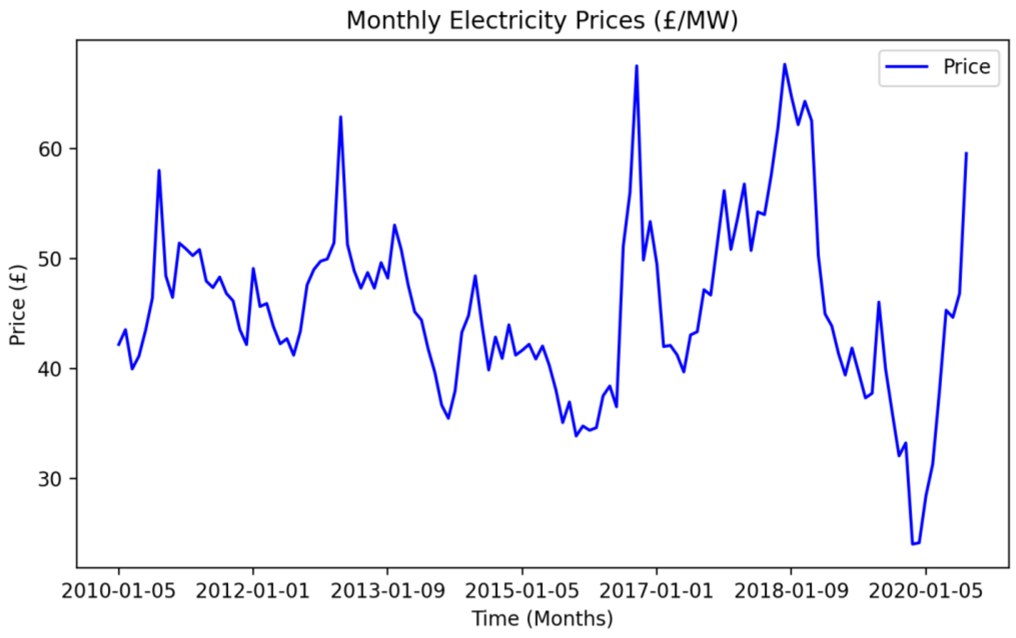


Fig 1.

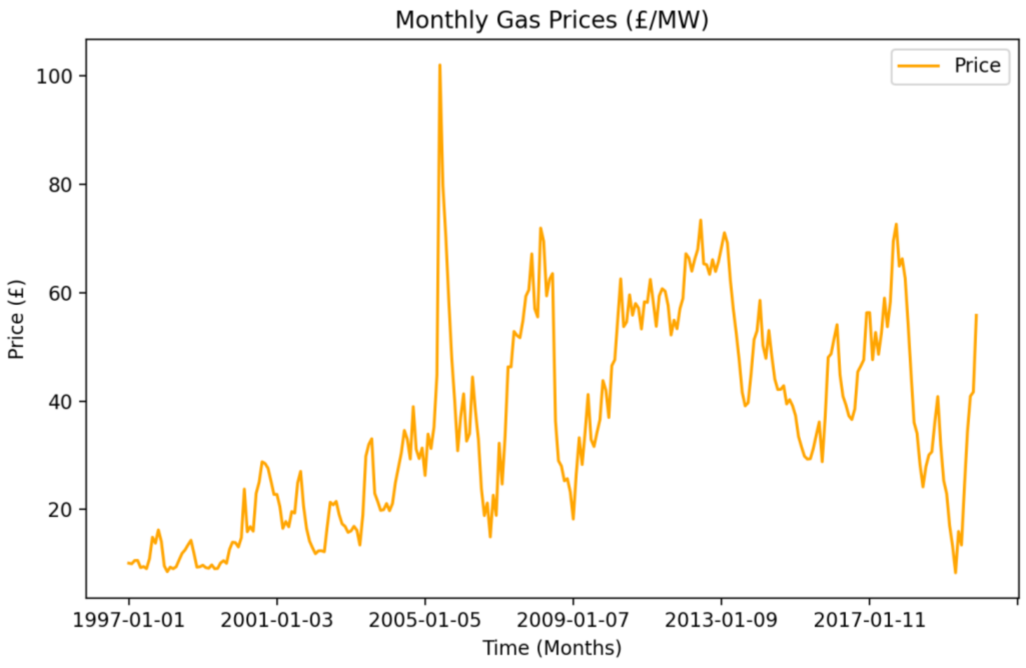
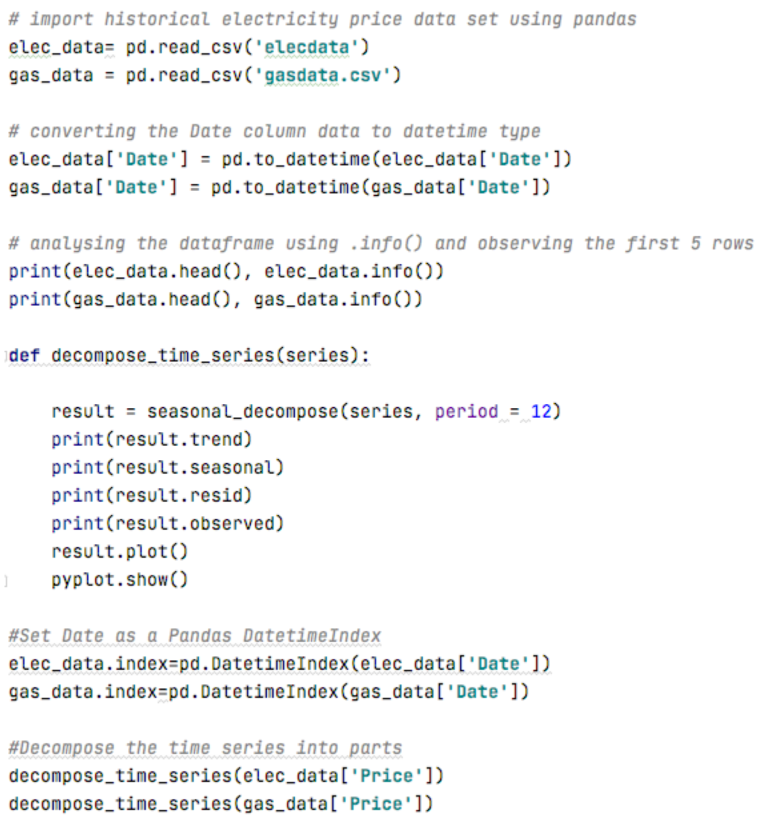


Fig 2.

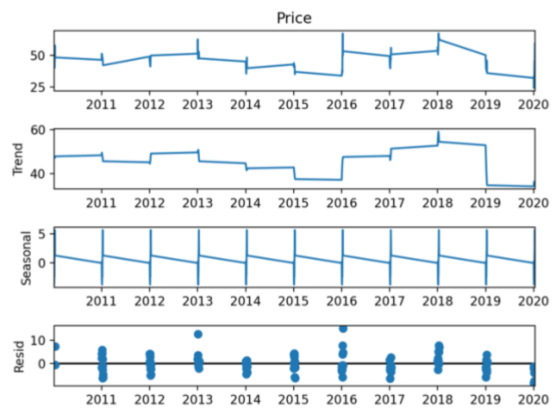
The graphs allow us to gain an understanding of how the prices have moved over time. We can see from a quick analysis that both sets of prices follow a similar pattern when looking from their common time frame (2010->). Both graphs inform us that there is evident volatility in the energy markets while also showing some boundary levels as the prices auto correct after price bounces and dips.

3.3 Seasonality Analysis

As outlined by Kenton (2020), seasonality is a characteristic of time-series where the data has predictable and somewhat regular fluctuations that repeat year over year. Energy price time series often see significant seasonality due to cyclical demand patterns. To decompose both time series the seasonal_decompose function from the statsmodels.tsa package was used. After the packages were imported, a user defined function was created. This function analysed the trend, seasonality, residuals and observed data. Both 'Date' columns were then set as pandas DatetimeIndex and passed through the defined function. We can see from both time series decomposition calculations, that WE prices and NG prices experience seasonality.

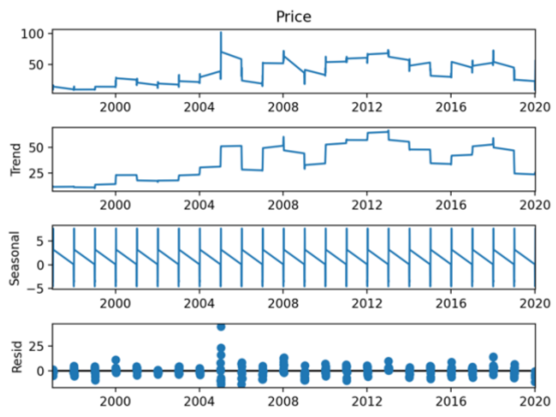


```
# import historical electricity price data set using pandas
elec_data= pd.read_csv('elecdata')
gas_data = pd.read_csv('gasdata.csv')

# converting the Date column data to datetime type
elec_data['Date'] = pd.to_datetime(elec_data['Date'])
gas_data['Date'] = pd.to_datetime(gas_data['Date'])

# analysing the dataframe using .info() and observing the first 5 rows
print(elec_data.head(), elec_data.info())
print(gas_data.head(), gas_data.info())

def decompose_time_series(series):

    result = seasonal_decompose(series, period = 12)
    print(result.trend)
    print(result.seasonal)
    print(result.resid)
    print(result.observed)
    result.plot()
    pyplot.show()

#Set Date as a Pandas DatetimeIndex
elec_data.index=pd.DatetimeIndex(elec_data['Date'])
gas_data.index=pd.DatetimeIndex(gas_data['Date'])

#Decompose the time series into parts
decompose_time_series(elec_data['Price'])
decompose_time_series(gas_data['Price'])
```



Electricity Time Series Decomposition

Fig 3.



Natural Gas Time Series Decomposition

Fig 4.

The output of the seasonal decomposition function produces the two sets of plots above. We can see from the plots that both NG and WE have similar seasonality patterns. This provides us with a strong indication that both data sets will likely see similar movements and exhibit a positive relationship. To investigate this further we must compare both variables in the same plot and test their correlation.

3.4 Comparative Analysis and Correlation

The next step in determining the usefulness of NG prices as a proxy variable for WE is to determine their relationship, if one exists. Firstly, both df's were merged using .merge_ordered( ) method – WE df left, NG df right, on the 'Date' columns. Two suffixes were created also.

```
# merge elec data with gas data using .merge_ordered and suffixes
merged_data = pd.merge_ordered(elec_data, gas_sorted, on ='Date', suffixes= ('_elec_data', '_gas_data'))
print(merged_data.head())

# checking for missing values due to the different time spans of the datasets
print(merged_data.isnull().sum())

# dropping rows of merged dataset which contain missing values
merged_data= merged_data.dropna()
print(merged_data)
```

```
        Date  Price_elec_data  Price_gas_data
0 1997-01-01              NaN          10.120
1 1997-01-02              NaN          10.000
2 1997-01-03              NaN          10.600
3 1997-01-04              NaN          10.625
4 1997-01-05              NaN           9.300
Date                0
Price_elec_data   160
Price_gas_data      0
```

```
          Date  Price_elec_data  Price_gas_data
160 2010-01-05            42.18           43.79
161 2010-01-06            43.53           42.02
162 2010-01-07            39.95           36.98
163 2010-01-08            41.12           46.53
164 2010-01-09            43.50           47.61
```

From the first view we can see missing values in the 'Price_elec_data' column for every row from 2010 where the data starts back to the first row (1997) where the gas data starts. Using the

.dropna( ) command we remove all rows from the df with missing values – this sorts the df with both price columns starting on the same date through to the final date. Now that both columns are in the same df and span the same time period, we can plot them together to analyse their relationship.

Using the matplotlib package again we set an ax plot for the subplots function. Using ax.plot( ) each column of the merged data df is then called in on the same ax ('Price_elec_data' and 'Price_gas_data'). Creating a list of labels and specifying the xticks data points allowed for a custom set of xtick labels for a cleaner look. A legend and other labels were included also.

```
merged_data = pd.read_csv('mergeddata')
print(merged_data.head())
print(merged_data.tail())
fig, ax = plt.subplots()
ax.plot(merged_data['Date'], merged_data['Price_elec_data'],label ='Electricity Price')
ax.plot(merged_data['Date'], merged_data['Price_gas_data'], label='Gas Price')
labels = ['2010','2011','2012','2013','2014','2015','2016','2017', '2018','2019','2020']
plt.xticks(['2010-01-05','2011-01-01','2012-01-01','2013-01-01','2014-01-01','2015-01-01','2016-01-01','2017-01-01', '2018-01-01','2019-01-01','2020-01-01'], labels)
plt.xlabel('Time (Months)')
plt.ylabel('Price (£)')
plt.title('Correlation Analysis')
ax.legend()
plt.legend(loc='upper right')
plt.show()
```

We can see from the graph below that there exists a clear positive relationship between the NG prices and WE prices. Both prices move in very similar fashion and exhibit the similar price reactions and bounces in variance. This gives us good confidence that NG prices can in fact be used a successful proxy variable in WE price forecasts. Bellemare (2015) resounds that in order for a variable to be a good proxy, it must have a strong correlation, not necessarily a linear or positive, with the variable of interest. This is what is investigated next.

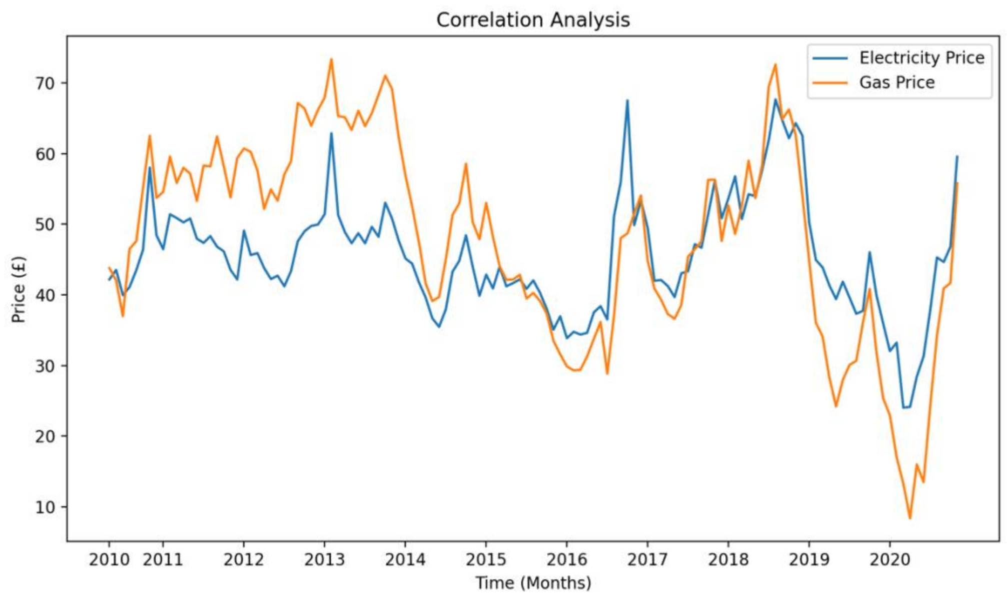
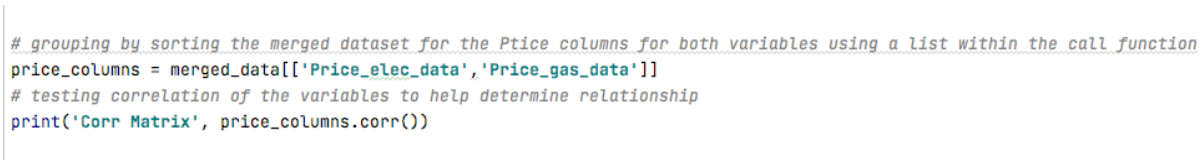
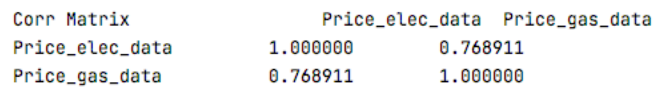


Fig 5.

The correlation of NG prices and WE prices was calculated using the .corr( ) command in pandas. Both price columns in the merged data df were assigned to price_columns. Then the .corr( ) command was executed on 'price_columns'. This produced a correlation coefficient of 0.768911. This confirms a strong positive relationship between the two variables. Any correlation coefficient of 0.4 – 0.7 is considered a moderate positive relationship, while a correlation coefficient of 1.0 is a perfect positive correlation. As such, a coefficient-0.768911 indicates that both variables move in tandem.



```
# grouping by sorting the merged dataset for the Ptice columns for both variables using a list within the call function
price_columns = merged_data[['Price_elec_data','Price_gas_data']]
# testing correlation of the variables to help determine relationship
print('Corr Matrix', price_columns.corr())
```



```
Corr Matrix                 Price_elec_data  Price_gas_data
Price_elec_data        1.000000        0.768911
Price_gas_data         0.768911        1.000000
```

4. Conclusion

This paper has examined in detail the relationship between NG prices and WE prices and has shown that NG can in fact be considered a useful proxy variable in WE price forecasting. To summarise, the main insights drawn from the visual analysis e: (1) there is a definitive presence of volatility and variance in the energy markets, showing clear price fluctuations across time – [fig 1,2]; (2) from both individual plots we can see a clear price range, the volatility moves in a banded range over time – [fig 1,2]; (3) both NG and WE exhibit seasonality in their price variance, an indication of a potential relationship – [fig 3,4]; (4) the interpretation of the comparative plot suggests a definitive, positive relationship between NG and WE price variables, clear indication of positive correlation – [fig 5]; (5) fig 5 also shows a slight divergence in the relationship during the period 2011 – 2014, this could potentially be attributed to varying fuel generation sources, this could be a point for further investigation – [fig 5].

After creating a reasonable assumption about the relationship between the two variables, through the various visual analysis tools above, this paper then looked to test this to confirm. This is when the correlation coefficient of the variables was calculated using the .corr( )function. We retrieve a correlation coefficient of 0.76811. This coefficient is indicative of a positive strong correlation between NG and WE prices. Based on our analysis, we can conclude that NG prices can be considered a useful proxy variable in different forecasting methods when forecasting WE prices.

5. Bibliography

Barchart.com. 2021. *Barchart.com | Commodity, Stock, and Currency Quotes, Charts, News & Analysis*. [online] Available at: <https://www.barchart.com/> [Accessed 15 April 2021].

Bellemare, M., 2015. *'Metrics Monday: Proxy Variables*. [online] Marc F. Bellemare. Available at: <https://marcfbellemare.com/wordpress/11115> [Accessed 23 April 2021].

Evans, S., 2019. *Analysis: UK electricity generation in 2018 falls to lowest level since 1994 | Carbon Brief*. [online] Carbon Brief. Available at: <https://www.carbonbrief.org/analysis-uk-electricity-generation-2018-falls-to-lowest-since-1994> [Accessed 26 April 2021].

Kenton, W., 2020. *What Is Seasonality*. [online] Investopedia. Available at: <https://www.investopedia.com/terms/s/seasonality.asp#:~:text=What%20Is%20Seasonality%3F,is%20said%20to%20be%20seasonal.> [Accessed 24 April 2021].

Kernan, R., Liu, X., McLoone, S. and Fox, B., 2017. The Effect of Wind Generation on Wholesale Electricity Prices in Ireland. *Queen's University Belfast*,.

Nordpoolgroup.com. 2021. *See what Nord Pool can offer you.*. [online] Available at: <https://www.nordpoolgroup.com/> [Accessed 17 April 2021].

Oxfordenergy.org. 2015. *The Role of Gas in UK Energy Policy*. [online] Available at: <https://www.oxfordenergy.org/wpcms/wp-content/uploads/2015/07/NG-100.pdf> [Accessed 26 April 2021].