UCD Michael Smurfit Graduate Business School

House Prices in the Seattle area (USA) – Studying predictor relationships through Multiple Linear Regression

Masters: Financial Data Science

Student Name: Evan Mc Garry

Student Number: 22206023

UCD Michael Smurfit Graduate Business School

Report - Part 1 & 2

Introduction

Why the Seattle Housing Market?

Seattle consistently ranks as one of the top 10 best cities in the USA for jobs due to the bustling economy that has been created by the tech-giants that have made it their home (examples include Amazon, Microsoft, Nintendo, Google, Apple and many more) as well as its inextricable link to Starbucks. Therefore, people living in Seattle will be very focused on their careers which can lead to the development of an interesting housing market where the peculiar needs of a talented and in-demand workforce must be met. The house price data used is in the Seattle area at 01/01/2019.

With this in mind, the general assumption is that if we model 'House Price' on a regressor such as 'Square Foot of Living Space' the variability of house prices will increase as living space increases. This is because extra living space often denotes the inclusion of 'luxury accessories' to the house such as pools, autonomous built-in household appliances, waterfront views or guest rooms which will inflate the price outside of what most people would see as the 'necessities'.

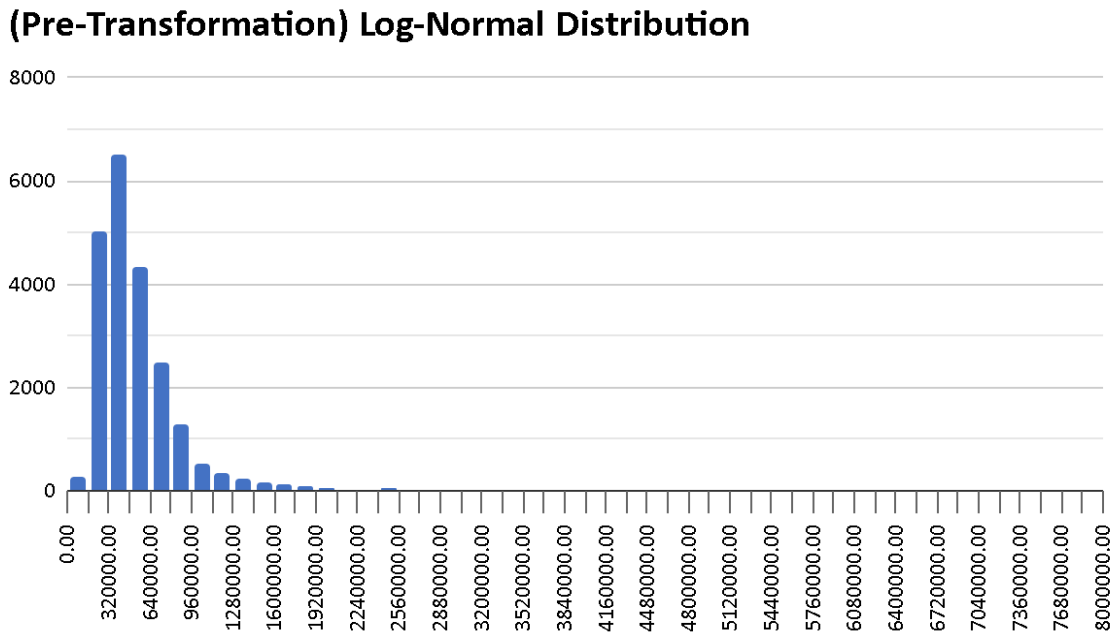
The inclusion of any 'luxury parameters' would theoretically help to more accurately predict house prices – but this report will focus on using the more 'standard' predictors to determine whether the housing market of this tech-centric city still adheres to general trends as seen elsewhere. In the unlikely event that the model is poor and the key predictors are statistically insignificant then this will allow us to draw interesting conclusions about the 'Seattle Area Housing Market' and its resident population.

The Method Used

This report will be constructed on the basis of a 'Multiple Linear Regression function' built in MatLab. The dependent variable is 'House Price' and is regressed on 8 independent variables. When discussing the economic interpretation of the result it must be established that the dependent variable 'House Price' has undergone a natural logarithmic transformation whereas all regressors are in 'level' form which will make this a '*log vs. level*' multiple linear regression. The estimated coefficients will therefore no longer be the '*marginal effect*' of a change in the dependent variable with respect to the change in the independent variable – it will become the **'growth rate**' (known as ***semi-elasticity***).

Why use the Natural Logarithmic Transformation?

The natural logarithmic transformation was applied since it is a monotonic transformation and reducing the scale of the 'House Prices' improved the model since the regressors were measured on a significantly smaller scale. Additionally, upon examination of the house prices, they exhibited a log-normal distribution:



This means that significant outliers were present on the right-hand side which posed a threat to the model in the sense that heteroskedasticity was not just likely

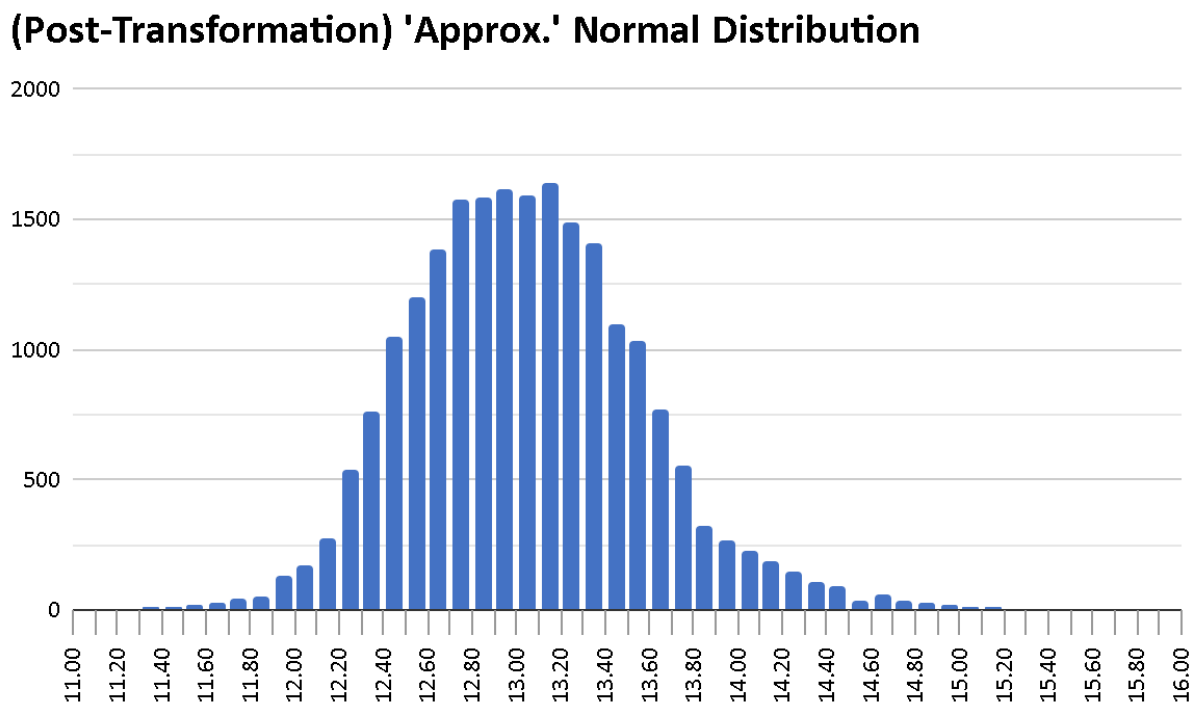
but unavoidable and present to a large degree. The skewness of the dependent variable was likely to cause issues once diagnostic tests on residuals were performed.

Result of the Transformation

The result of the transformation was a '*more normal*' distribution of our dependent variable, as well as a better scaled matrix which our regression analysis will be based on. If the matrix is poorly scaled then MatLab (our chosen software) would produce the error:

> **Warning: Matrix is close to singular or badly scaled. Results may be inaccurate.**
> **RCOND = NaN**

The resulting distribution was much more 'normal' and while this is not necessary to run the regression, it will undoubtedly improve the model:



The resulting distribution is not perfect, just by inspection we can see positive skewness as well as a degree of excess kurtosis. However, it is much more normal than the pre-transformation dependent variable.

As mentioned, the interpretation of the estimated coefficients will change and whilst this is a drawback of the transformation, the new interpretation is not as cumbersome as other transformations. The new estimated coefficient is:

$$\widehat{\beta}_i = \frac{\delta Y_t}{\delta X_{it}} \times \frac{1}{Y_t}$$

Therefore, we have the marginal effect multiplied by the inverse of the observed dependent variable. This will give us **semi-elasticity** or **growth rate** and this has the effect of converting changes in the dependent variable to being presented in percentage terms (%) while changes in the independent variable remain in their respective units. With this kept in mind, the natural logarithmic transformation will prove very useful and not significantly hinder our interpretation.

What are the Predictors (Regressors)?

The independent variables used are as follows:
[Note: x1 will be a constant to be mathematically consistent]

x2	Number of Bedrooms	Key predictor* (‘Necessary’)
x3	Number of Bathrooms	Key predictor* (‘Necessary’)
x4	Living Area (sq.ft)	Key predictor* (‘Necessary’)
x5	Year Built	‘Niche’ predictor
x6	Basement Area (sq.ft)	‘Niche’ predictor
x7	Latitude	Key predictor* (‘Necessary’)
x8	Longitude	Key predictor* (‘Necessary’)
x9	Grade	‘Niche’ predictor

Why these Predictors & Expectations?

(1) X2:

- The '*number of bedrooms'* is necessary because generally fewer bedrooms means a cheaper house as this will typically imply a *smaller living area*.
- If there are a greater number of bedrooms then this implies that the house is a 'family home' which will require a larger living area and the price should be higher.
- This can also double up as a 'niche predictor' because a greater number of bedrooms could mean that there are 'guest rooms' which would mean someone is living outside of what could, somewhat unconvincingly, be classed as 'necessary'.

(2) X3:

- The necessity of '*number of bathrooms'* follows similar reasoning as the first – more bathrooms will imply a greater living space and the use of the house as a 'family home'.
- Additionally, a higher number of bathrooms hints at the idea of en-suites which will typically be found in the more expensive houses.

(3) X4:

- The necessity of '*living area*' is paramount as bigger houses always command a higher price. This is the true 'key predictor'.

(4) X5:

- The '*year built*' is a hybrid predictor because there can be 2 extremes – old houses can be very expensive due to their appearance and low supply and brand new houses can also attract large prices because they can be fitted with modern technology that would not otherwise be available.
- However, the more likely interpretation would be that older houses are less expensive due to poorer facilities and structural integrity.
- The outcome of this estimated coefficient will be highly interesting, making it overall a luxury (niche) predictor.

(5) X6:

- The '*basement area*' is a 'luxury predictor' because basements are often used for storage or perhaps the pursuit of a hobby that needs space. Additionally, few houses actually have basements.

(6) X7 & X8:

- '*Latitude*' and '*Longitude*' are intertwined and are extremely necessary because they account for 'distance from the city'. The city is where people will work and being nearer to the city, and the workplace by extension, is seen as necessary for a better quality of life.
- Many people could perhaps see this as being equally important as '*living area*' so this is a crucial predictor.

(7) X9:

- The '*Grade*' is an important consideration as it provides a scale for 'quality of construction' of the house. This will help account for custom-built houses that will command a massive fee.
- However, the construction of houses these days generally conforms to a high standard by law and as such this is a 'niche' predictor.
- The scale ranges from 1-13 with 1-3 being those that fall short of minimum building standards and 13 being 'mansion level, custom designed and built'. [For more information see "Building Grade" in: <eSales - King County>].

UCD Michael Smurfit Graduate Business School

Part 1

The first part of this report is to show the structure of the function that was used to obtain the necessary outputs that will be discussed in detail in **Part 2** of this report. The function was created using the code that will be discussed step-by-step in Part 2 of this report and as such this is only meant to show the structure of the function. Any and all detailed discussion will be reserved for **Part 2** of this report:

"Beta OLS Function"

```
function
[Beta_hat,CI,t_stat,p_value_t,R2,AdjR2,F_statistic,Fp_value,Skewness,Kurtosis,J
B,JBcrit,H,BP_stat,bpcrit,pvalue_BP,DW,VIF] = BetaOLSFunction(X,Y)
% Data
[data,text]=xlsread('test_data.xlsx');
data1 = data(:, :);
names = text;
Y = data1(:, 1);
X = data1(:, 2:end);[T,N] = size(X);
X = [ones(T,1) X];
K=size(X,2);
df = T-K;
% (a) OLS Estimators
[Beta_hat] = (X'*X)\X'*Y
% (b) Confidence Intervals
Yhat = X*Beta_hat;
residuals = Y - Yhat;
sigma_r = (residuals.'*residuals)/(T-K);
varbeta = sigma_r.*inv(X'*X);
stderr_beta = sqrt(diag(varbeta));
% Calculate Confidence Interval (CI)
CI = [Beta_hat - stderr_beta.*1.96, Beta_hat + stderr_beta.*1.96]
% (c) Statistical Significance
t_stat = abs(Beta_hat./stderr_beta)
p_value_t = tcdf(t_stat, T-N,"upper")*2
% (d) R^2 and Adjusted R^2
Ybar = mean(Y);
R2 = sum((Yhat-Ybar).^2)/sum((Y-Ybar).^2)
AdjR2 = 1-((T-1)/(T-K)*(1-R2))
% (e) F-statistic
regressors_remained = 1;
m = K-regressors_remained;
F_statistic = (R2/(m))/((1-R2)/(T-K))
Fp_value = 1 - fcdf(F_statistic,m,T-K)
% (f) Plot of the Fitted Model
% Plot of Fitted Model
```

```
figure(1)
coefficients = polyfit(Yhat,Y,1);
xFit = linspace(min(Yhat),max(Yhat),2000);
yFit = polyval(coefficients,xFit);
subplot(1,2,1)
plot(Yhat,Y,'b*')
hold on;
plot(xFit,yFit,'r-','LineWidth',2);
grid on;
xlabel("Estimated House Prices (giving Line of Best Fit)")
ylabel("Observed House Prices")
title("Plot of the Fitted Model")
hold off
subplot(1,2,2)
scatter(Yhat,residuals)
xlabel("Estimated House Prices")
ylabel("Residuals")
title("Residuals Scatter Plot")
hold off
% (g) Diagnostic tests for residuals
% Normal Distribution
figure(2)
subplot(1,2,1)
qqplot(residuals)
title('Normal Q-Q');
ylabel('Residuals');
subplot(1,2,2)
histogram(residuals);
title('Histogram of Residuals');
xlabel('Residuals'); ylabel('Frequency');
hold off
Skewness = skewness(residuals)
Kurtosis = kurtosis(residuals)
JB = T*((skewness(residuals)^2)/6 + ((kurtosis(residuals)-3)^2)/24)
JBcrit=chi2inv(0.95,2)
if JB > JBcrit      % H = 1; reject H0
   H = 1
elseif JB < JBcrit  % H = 0 means Normal Distribution present
   H = 0
end
% Heteroskedasticity
RSS = (residuals.'*residuals);
sigmaeps = (1./T).*RSS;
epsnew = (residuals.^2)./sigmaeps - 1;
Beta_BP = inv(X.'*X)*X.'*epsnew;
epsfitted = X*Beta_BP;
BP_stat = sum(epsfitted.^2)./2
bpcrit = chi2inv(0.95,m)
```

```
pvalue_BP = 1-chis_prb(BP_stat,m)
% Serial Correlation
DW=sum(diff(residuals,1).^2)./sum(residuals.^2)
% (h) Multicollinearity
R0 = corrcoef(X(:,2:end));
VIF_X = diag(inv(R0))';
VIF = array2table(VIF_X)
end
```

Fig(1.1)

With this function constructed, in the main script it is only necessary to run the code:

```
BetaOLSFunction(X,Y)
```

in order to achieve all of the outputs that have been described above in the function. All that is necessary for the function to run are the ‘X’ and ‘Y’ variables – therefore once the data has been imported with the provided code labelled (% Data) then the function will run successfully. The script for the function will be attached as a ‘.m’ file in addition to this visual aid provided in the report.

Part 2

This section of the report will incrementally introduce and analyse the code that was necessary to build the 'Beta OLS Function'. More importantly, the results from running the function will also be provided and analysed.

a) OLS Estimators

The first step is to construct the equation for OLS Estimation, and this is achieved through using matrix-vector format in MatLab. First, it is necessary to extract the independent and dependent variable vectors from the data. This was achieved in MatLab using the following code:

```
%% 1. Importing Data
% House-Price in the USA at 1/1/2019 (pre-cleaned) - using 8
Regressors
% Run this before running the 'OLS Regression Function'
[data,text]=xlsread('houseprice_data_2019.xlsx');
data1 = data(:, :);
names = text;
Y = data1(:, 1);
X = data1(:, 2:end);
[T,N] = size(X);
X = [ones(T,1) X];
K=size(X,2);
df = T-K;
```

Fig(2.1)

The 'X' matrix includes a column of ones which represents the constant and this is necessary to be mathematically correct. The 'Y'-vector is a column-vector of the 'Log House Prices'. 'T' represents the number of observations in the sample (21,613) and 'K' represents the number of coefficients, including the constant. The formula for the OLS estimators (Beta_hat) is constructed on the basis of 'X' and 'Y'-vector and will be shown in the code as being the inverse of the product of the *transpose* of 'X' and 'X' multiplied by the product of 'X' *transpose* and Y-vector.

The predicted values of the fitted model (Yhat) are then found by multiplying 'X' by the estimated coefficients. The residuals (epsilon) are the distance between the predicted and observed value of 'Y' and can be found mathematically by

subtracting the estimated 'Y' value from the observed 'Y' value. The code necessary to calculate these values is as follows:

```
% OLS Estimators

Beta_hat = (X'*X)\X'*Y

% Estimated 'Y'

Yhat = X*Beta_hat;

% Residuals

residuals = Y - Yhat;
```

Fig(2.2)

These 3 variables are crucial and it is also necessary to calculate the variance and standard error of the 'OLS Estimators' and the 'Residuals' which can easily be completed in MatLab with the following code:

```
% Variance & Standard Error of Residuals

sigma_r = (residuals.'*residuals)/(T-K); % Residual Variance
Estimator

r_stderr = sqrt(diag(sigma_r)); % Residual Variance Std. Error

% Variance & Standard Error of OLS Estimators

varbeta = sigma_r.*inv(X'*X); % Var/Covar Matrix

stderr_beta = sqrt(diag(varbeta)); % Standard Errors
```

Fig (2.3)

Results:

Only the OLS Estimators are provided here as the rest of the code will be necessary for further analysis in later parts of the report:

Regressors	Estimated β
X1 (C)	-43.83582
X2	-0.021922
X3	0.091361
X4	0.0001981
X5	-0.0041506
X6	-1.07E-06
X7	1.3073026
X8	-0.0072202
X9	0.19275

With our estimated coefficients (**β**), we can begin interpreting our results.

Economic Interpretation

The first form of interpretation we shall consider is the '**economic interpretation**' of our Betas, as the statistical interpretation will require further code which will be introduced later. As discussed, this is a "log vs. level" regression so we will consider the **semi-elasticity** when discussing the meaning and importance of each value of Beta:

1. X1 (Constant)

X1 (C)	-43.83582

The presence of a negative constant is not only *not* a cause for concern, it's a very welcome sight. The constant represents the expected (log) value of the dependent variable if all of the **regressors** are equal to 0. Unsurprisingly, this is a negative number since a house with no living area is no longer a house!

2. X2 (No. of Bedrooms)

X2	-0.021922

The negative Beta here is quite surprising given the initial expectations for this regressor set out in the introduction to the report. This means that when we *increase* the number of bedrooms by 1 unit, the price of the house will *decrease* by 2.2%. This was a very surprising outcome and could be due to a number of factors but it seems that the people of Seattle prefer solitude and will pay extra for it!

3. **X3 (No. of Bathrooms)**

X3	0.091361

The positive Beta here is in line with initial expectations, albeit it is quite a large coefficient. This means that when we increase the number of bathrooms by 1 unit, the house price will increase by roughly 9%. This is unsurprising since most houses will be fitted with an adequate number of bathrooms and any additional bathrooms are most likely ensuite which imply a more expensive house since an en-suite is a luxury and not strictly necessary. It could also be due to the fact that extra 'living area' will require extra bathrooms for ease of access.

4. **X4 (Living Area [sq.ft])**

X4	0.0001981

Crucially, there exists a positive Beta for our key predictor. The Beta is very small but this is to be expected due to the scale of 'living area' being large in the form of square feet. This means that an increase of 1 unit of square foot living area will result in a house price increase of 0.02%. This is unsurprising since an increase of 1 square foot of living area would barely even be noticeable – however the positive sign of the Beta is crucial since it affirms our belief that the larger the living area of the house the more expensive it will be.

5. **X5 (Year Built)**

X5	-0.0041506

This regressor was an interesting one for consideration as there were a few ways of interpreting how house prices might react to the year the houses were built. Quite surprisingly there exists a negative relationship which means that an increase of 1 year in the year the house was built will cause the house price to decrease by 0.4%. This means older houses are generally more expensive – this could mean that the

Seattle workforce have retro-tastes or it could be something as simple as older houses would generally have been built closer to the city and occupy that prime land. As we will see this may well be the case since there exists a highly positive relationship with latitude – however this is just speculation and would require further research to determine the veracity of this statement.

6. X6 (Basement [sq.ft])

X6	-1.07E-06

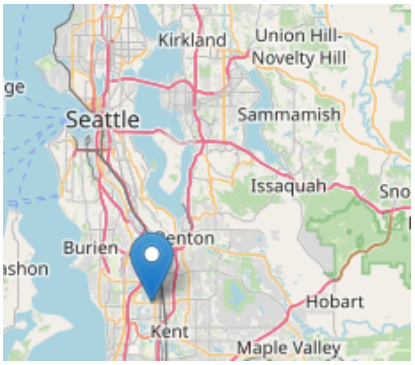
This was a niche predictor included to determine whether basements are popular and sought after in Seattle. The answer is a resounding no – there exists a negative relationship which means an increase of 1 square foot of basement area will result in a house price decrease of a tiny fraction of a percentage. It can be concluded that basements are borderline irrelevant in the housing market in Seattle.

7. X7 (Latitude)

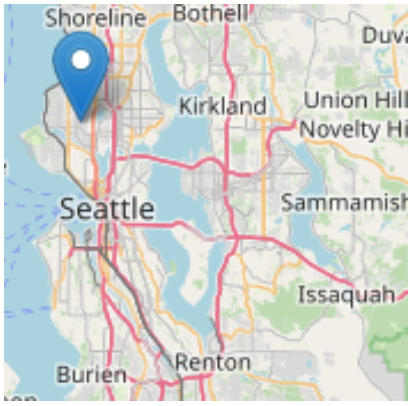
X7	1.3073026

This predictor was extremely necessary and produced an overwhelmingly positive result. However, in order to correctly interpret this result it is necessary to provide background information regarding latitude. We can say that the latitude for Seattle is at 47° which remains fixed – the changes in this variable are in the minutes and they typically range from 47**.2** to 47**.8** in the sample used for this regression. One 'minute' is roughly equal to about 1.15 miles. As the **minutes increase**, the location of the house becomes increasingly closer to the **Seattle city-centre** as I will demonstrate:

Latitude = 47.51, Longitude = –122.2



Latitude = 47.67, Longitude = –122.3



As we can see, an increase of about 1.6 minutes in Latitude caused the location of the house to change and become much closer to the city by moving north (as viewed from this picture). With such drastic changes caused by latitude it is unsurprising that the positive relationship is so massive with a 1 unit (**minute**) increase resulting in an increase in house price of about 131%! This seems to confirm the existing belief that the closer people living in Seattle are to the city, the more they are willing to pay for the property. This is a general trend not specific to Seattle but it exists to a very large degree in Seattle.

However, this must be discussed in tandem with longitude because as we see, longitude also causes a lateral movement in the location of the property (as viewed from this picture).

8. X8 (Longitude)

X8	-0.0072202

In a similar vein to latitude, the relationship with longitude is very unsurprising – given that the Seattle area generally has a longitude of –122. If we look at the previous subsection of latitude we saw that a decrease in longitude caused the location of the property to move laterally towards the city centre. Logically, as the minutes of longitude increase we will move gradually away from the city centre – however the relationship is much less severe than that of latitude. An increase of 1 unit (**minute**) of longitude results in a price decrease of 0.7%. This further backs

up the existing assumption that people in Seattle will pay more to live closer to their workplace.

9. **X9 (Grade)**

X9	0.19275

This 'niche predictor' has proved extremely useful and it exhibits a strong positive relationship with house price, which was to be expected. This predictor can undoubtedly help capture some of the outliers which can only be described as 'mansions' that command fees in the millions. A 1 unit increase in Grade will result in a 19% increase in house price. In Seattle it seems quality really matters!

b) Confidence Intervals

Next it is necessary to consider the individual *statistical significance* of the OLS Estimators, there are numerous ways of doing this but the first we will consider are the '**confidence intervals**'. The confidence interval can be described as an *interval* that contains the possible values of our OLS Estimators (Beta) and is subject to hypothesis testing:

- H_0; $\beta = 0$
- H_1; $\beta \neq 0$

If we fail to reject our **null hypothesis** (H_0) then our estimator is *statistically* ***insignificant*** as Beta can equal 0 in our sample. It is necessary at this point to establish our '**Level of Significance**' which is our alpha – for this report we will use a 5% level of significance (*i.e.* alpha = 0.05).

If the confidence interval ***includes*** **'0'** then we fail to reject H_0 at the 5% significance level. The formula for constructing the confidence interval in MatLab is as follows and it uses the standard error of Beta which we calculated previously [Fig(1.3)]:

```
%% 2. Confidence Intervals [Part 1(b)]
alpha = 0.05 % Level of Significance
t_critical = tinv(1-alpha/2,T-N); % Critical 't' = 1.96
CI = [Beta_hat - stderr_beta.*1.96, Beta_hat + stderr_beta.*1.96]
```

Fig(2.4)

Running this code will return a 9x2 matrix with the confidence intervals given between columns. The confidence intervals are for each regressor (including the constant 'x1') in descending order of x1-x9:

Regressors	Confidence Interval	
X1 (C)	-47.6835	-39.9882
X2	-0.02681	-0.01703
X3	0.08325	0.09948
X4	0.00018945	0.0002067
X5	-0.0043164	-0.003985
X6	-1.11E-05	8.91E-06
X7	1.2803	1.3343
X8	-0.03644	0.02200
X9	0.1874	0.1981

The important thing to look for is a change in positive or negative sign moving from the lower bound to the upper bound of the confidence interval. Unfortunately, not all of the regressors used are statistically significant. We have **2 statistically insignificant regressors** and these are:

- **X6** which is the **square foot** of the **basement**.
  This is unsurprising given that not many houses actually have basements and the extremely low impact it has on house price.
- **X8** which is the **longitude**.
  This is quite surprising, since latitude has such a huge impact on the house prices. However, one minute of longitude is roughly 0.9 miles which is less than latitude meaning the relationship is less severe (almost half the impact of a 1 unit increase of latitude).

H_0 cannot be rejected at the 5% significance level for these predictors and as such they are *statistically insignificant* since Beta can equal '0' in the sample.

c) Statistical Significance

It is also necessary to consider the statistical significance of each estimator individually using a different method. This is a similar method to the confidence interval since it considers each estimator individually and it arises from a ratio of a *normally distributed numerator* and a χ^2 distributed denominator – which means we use the '***t-statistic***' again to check statistical significance.

It is necessary to obtain an *unbiased estimator* for the **variance** of Beta and we have done this previously [Fig(2.3)]. The variance and standard error of the OLS estimator that we have already obtained through our MatLab code overcomes the issue of unbiasedness and we can readily plug this into the formula to obtain the **empirical 't-statistic'** which is given by the following code:

```
%% 3. Statistical Significance of Beta Coefficients [Part 1 (c)]
% H0 (null hypothesis): beta_i = 0 (for i=0,...,8)
% H1 (alternative hypothesis): beta_i ~= 0 (for i=0,...,8)
t_stat = abs(Beta_hat./stderr_beta) % empirical t-statistics
t_critical = tinv(1-alpha/2,T-N)
p_value_t = tcdf(t_stat, T-N,"upper")*2
```

Fig(2.5)

This code is extremely useful because it provides the empirical t-statistic and the critical value of t at the 5% level of significance with T – N degrees of freedom ('N' since we exclude the constant when computing the t-statistic). Typically if the empirical 't-statistic' is greater than the critical t-value then you reject H_0 but the **'p-value**' of the 't-statistic' is included in this code which does this manually – the **p-value** tells you how likely it is that your *data* could have occurred under the **null hypothesis**.

Therefore, if the **p-value** is *less than* 0.05 then the predictor is statistically significant at the 5% significance level:

Regressors	P-value 't' [0.95]
X1 (C)	0
X2	0

X3	0
X4	0
X5	0
X6	0.8330
X7	0
X8	0.6282
X9	0

We can see that the same result has occurred as that of the 'confidence interval'. The regressors **X6** and **X8** are statistically insignificant. A course of action would be to remove the more statistically insignificant predictor first (**X6 - basement**) and run the model again to see if all of the predictors are statistically significant. If not, then we could remove **X8** and run the model again.

d) Goodness of Fit

The 'goodness of fit' of a model is how **effective** of a **fit** the model is – *i.e.* how much of the data can be explained by the fitted model. There are two indicators to check here, 'R^2' and 'Adjusted R^2'.

R^2 & Adjusted R^2

This is known as the **coefficient of determination** and this is the first number we check to see if a regression is good. Typically, a regression with an R^2 value of 0.6 and above is considered 'good' – however this can be entirely subjective. Before showing the code for R^2 it is necessary to introduce the idea of 'Adjusted R^2' because this is necessary for comparing models.

R^2 can be used to compare models with the *same dependent variable* but the issue is that R^2 is biased and will increase when additional predictors are added, regardless of whether or not the predictors added are significant or not! Therefore, in order to accurately compare models, we compute the 'Adjusted R^2' by removing the bias from R^2:

```
%% 4. Goodness of Fit [Part 1 (d)]
% R-squared
Ybar = mean(Y);
R2 = sum((Yhat-Ybar).^2)/sum((Y-Ybar).^2)
% Adjusted R-squared
AdjR2 = 1-((T-1)/(T-K)*(1-R2))
```

Fig(2.6)

This code shows how we calculate the 'goodness of fit' of the model and it is a simple calculation to do on MatLab after running the regression as you require the predicted values of Y. For our model, the goodness of fit is:

R^2	Adjusted R^2
0.7384	0.7383

This is quite a good model as it explains ~74% of the observations – however with house prices it can become even more accurate since house prices are quite stable under normal economic conditions which were present in 2019. There is room for improvement in the model despite a promising R^2.

e) F-statistic

The F-statistic is introduced at this point since we are testing more than 1 restriction. The F-statistic is used to test the *statistical significance* of all the slopes jointly – which means we want to test the impact of the regressors. Since we are discussing slopes, the constant (X1) will not be considered in the null hypothesis. The code is as follows:

```
%% 5. F-statistic
% H0; Beta_2 = Beta_3 = Beta_4 = Beta_5 ... Beta_10 = 0]
% H1; Beta_i ~= 0 (for i=2,...,10]
% Restricted Regression
regressors_remained = 1;
m = K-regressors_remained; % the number of restrictions
% Empirical 'F-Statistic'
F_statistic = (R2/(m))/((1-R2)/(T-K))
% P-Value
Fp_value = 1 - fcdf(F_statistic,m,T-K)
```

Fig(2.7)

As we see, the null hypothesis (H_0) is that the slopes (estimated coefficients) of all the regressors equal 0 – this provides 'm' number of restrictions which is used in the calculation of the **empirical F-statistic** and 'm' excludes the constant as the 'constant' regressor must remain in the **restricted model** to run the **auxiliary regression**. However, the formula used to calculate our empirical F-statistic only uses R^2 and since the restricted model will have a $R^2 = 0$ due to it consisting of a constant and error term we are able to calculate the F-statistic using our original R^2 and the necessary degrees of freedom for each χ^2 as shown above.

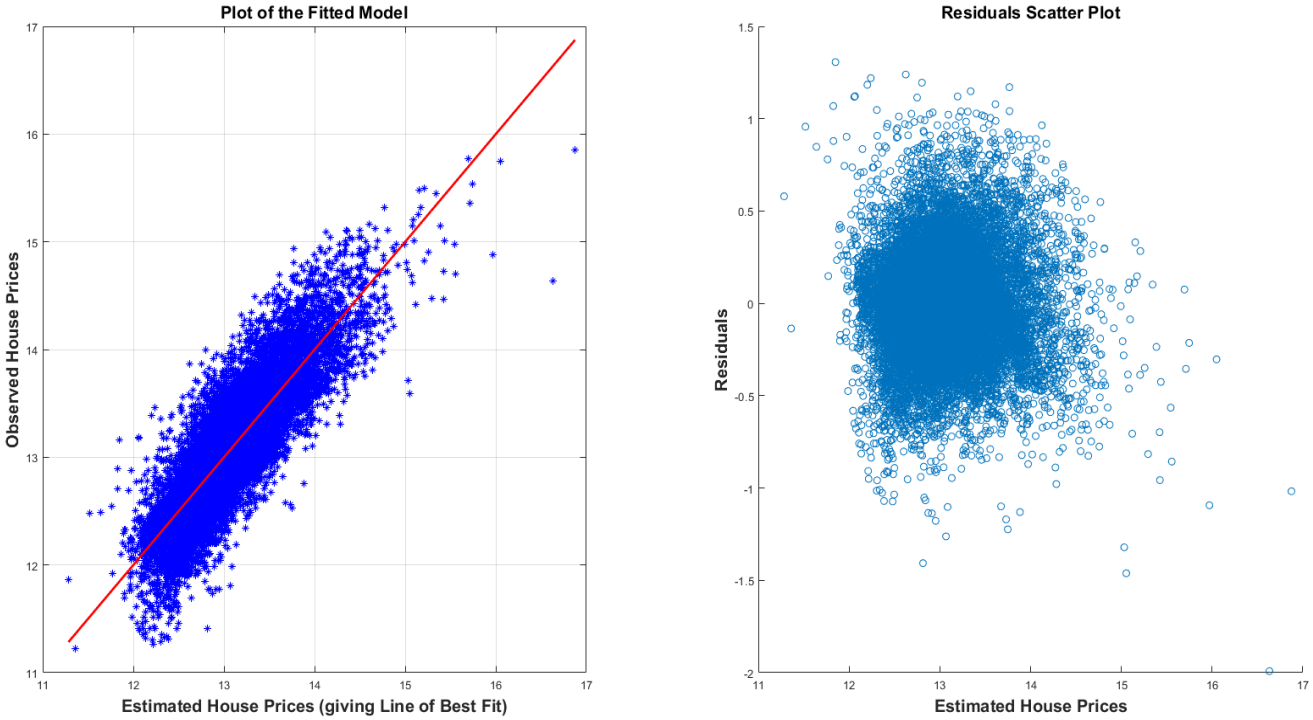
However, rather than compare the empirical and the critical values of 'F-statistic' we can use the **p-value** in the same way as before. If the **p-value** is *less than* 0.05 then we can *reject* the null hypothesis and conclude that the slopes are *jointly significant*. The 'F-statistic' and 'p-value' results are as follows for our model:

F-Statistic	P-Value 'F' [0.95]
7662	0

The high 'F-statistic' is due to our high R^2 and large number of observations (21,613). The **P-value** is 0 which means that the slopes are jointly statistically significant.

f) Plot of the Fitted Model

Now it is necessary to visualise our model – the plot of the fitted model will allow us to see how well the model fits the observations of 'Y' that were collected. The R^2 is quite high so the line of best fit is likely to have an even distribution of observations above and below the line. This also allows us to see the *residuals* in a graphical form as was described previously in the report – rather than just calculating the mathematical form of the residuals. For this reason, the report will show subplots of the fitted model as well as the residuals in order to make some preliminary comments about the residuals:



Fig(2.8)

- As expected, the fitted model is quite evenly distributed, but not perfectly so which gives us an indication as to how the residuals will look. The presence of outliers is also apparent which is unavoidable given the disparity between the price of 'mansions' and normal houses.
- The residuals scatter plot is interesting and provides us with some information – it is unlikely that our residuals are homoskedastic. The variance is not constant as is seen by the spreading out of the residuals in the

plot, which means there is going to be heteroskedasticity present in the residuals.

- The issue likely lies within the dataset, because the range of values of house prices can be quite drastic – the logarithmic transformation undoubtedly helped with this but it does not eliminate the presence of heteroskedasticity. This is a problem for the model – but this will be discussed further later in the report.

The code used to generate the subplots seen above is as follows:

```
figure(1)
coefficients = polyfit(Yhat,Y,1);
xFit = linspace(min(Yhat),max(Yhat),2000);
yFit = polyval(coefficients,xFit);
subplot(1,2,1)
plot(Yhat,Y,'b*')
hold on;
plot(xFit,yFit,'r-','LineWidth',2);
grid on;
xlabel("Estimated House Prices (giving Line of Best Fit)")
ylabel("Observed House Prices")
title("Plot of the Fitted Model")
hold off
subplot(1,2,2)
scatter(Yhat,residuals)
xlabel("Estimated House Prices")
ylabel("Residuals")
title("Residuals Scatter Plot")
hold off
```

Fig(2.9)

As we have seen, this code runs automatically in the function without needing to specify it as an output of the function.

g) Diagnostic Tests for Residuals (Normality, Heteroskedasticity & Serial Correlation)

This section will involve running diagnostic tests on the residuals that we have seen above. While we have identified that heteroskedasticity likely exists, it is necessary to employ a statistical test to confirm this. Similarly, tests will be conducted to check if the residuals are ***normally distributed*** and ***serially correlated***. Ideally, we would like our residuals to exhibit a *normal distribution* and to *not* be **serially correlated**.

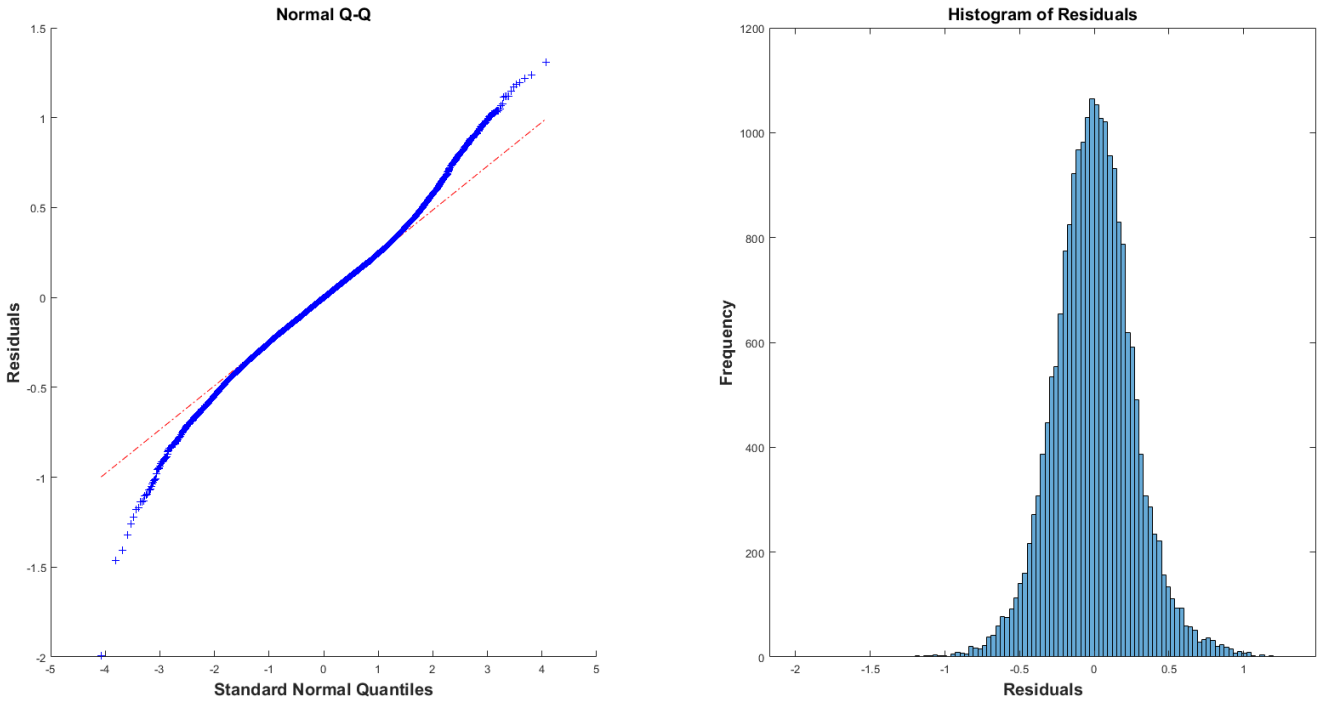
(i) Normal Distribution

Firstly, we can visually examine the residuals to see if they are normally distributed by using a **Q–Q plot** and a **histogram**. These two subplots were produced using the following code:

```
figure(2)
subplot(1,2,1)
qqplot(residuals)
title('Normal Q-Q');
ylabel('Residuals');
subplot(1,2,2)
histogram(residuals);
title('Histogram of Residuals');
xlabel('Residuals'); ylabel('Frequency');
hold off
```

Fig(2.10)

Which produced the following (next page):



Fig(2.11)

- The **Q–Q Plot** above shows characteristics of '*over-dispersed data*' which implies that there is an increased number of outliers in the dataset. This is characteristic of a 'Laplace Distribution' and is also known as a **leptokurtic distribution** which has **positive excess kurtosis**.
- Looking at the **histogram** of the residuals, it seems likely that we have a **leptokurtic distribution** as we have slightly fatter tails and a high peak of the distribution.

In order to determine whether or not the statements above hold, we need to introduce 2 important moments of a distribution. These are **skewness** and **kurtosis** – these are important because they should equal 0 and 3 respectively for a **normal distribution** and also because they are used to calculate the **test statistic** of the **"Jarque-Bera Test"** which we will employ to check the distribution of our residuals and give a definite answer – Jarque-Bera is also very popular because it can be used for large sample sizes which is very useful for this report.

Firstly, we will calculate **skewness** and **kurtosis** with the following simple code:

```
Skewness = skewness(residuals)
Kurtosis = kurtosis(residuals)
```

Fig(2.12)

After running this we get the following results:

Skewness	Kurtosis
0.091	4.1932

Skewness seems to be approximately equal to 0, but we have **excess kurtosis** of roughly 1.2. This backs up the idea that we have a **leptokurtic distribution** and *not* a **normal distribution**.

The **Jarque-Bera Test** will likely confirm our suspicions, and the test statistic as well as it's H_0 and H_1 are listed in the code as follows:

```
% Jarque-Bera Test
% H0; We assume that residuals are Normally Distributed
% H1; We assume that residuals are not Normally Distributed
Skewness = skewness(residuals)
Kurtosis = kurtosis(residuals)
JB = T*((skewness(residuals)^2)/6 + ((kurtosis(residuals)-3)^2)/24)
JBcrit=chi2inv(0.95,2)
if JB > JBcrit      % H = 1; reject H0
   H = 1
elseif JB < JBcrit  % H = 0 means Normal Distribution present
   H = 0
end
```

Fig(2.13)

Therefore, we see that the JB-statistic is a χ^2 distributed statistic with 2 degrees of freedom. Instead of **p-value**, the code manually returns the value of 'H' which is a binary variable with a value of 1 meaning that we **reject H_0** and conclude that the residuals are not normally distributed at the 5% significance level. Conversely, if H is returned as a value of 0 then we **fail to reject H_0** and we conclude that the residuals are normally distributed.
The above code produces the following results:

JB	JB Critical Value	H
1311.9	5.9915	1

Therefore, we can conclude that the **residuals** are *not* normally distributed – we have a **leptokurtic distribution** due to having **positive excess kurtosis**. The skewness may also be present, but it will cause only a slightly positive skewed distribution since it is very close to 0.

(ii) Proportional Heteroskedasticity

We are going to use the "**Breusch-Pagan Test**" to check for Heteroskedasticity – this will be sufficient because we have not squared our variables, or created cross-terms. This test involves running an auxiliary regression with the variance of the residuals as the dependent variable. The auxiliary regression will look like:

$$\hat{\varepsilon}_t^2 = \delta_1 + \delta_2 X_2 + \ldots + \delta_9 X_9 + \upsilon_t$$

This will allow us to test **proportional heteroskedasticity** (which tests whether the **heteroskedasticity** depends on the *regressors*). The following code shows how the auxiliary regression is set up, the test statistic calculated and produces a **p-value** for simple interpretation:

```
%% 8. Residuals Diagnostic Test for Heteroskedasticity [Part 1 (g)]
% Breusch-Pagan Test
% H0; Delta2 = Delta3 ... Delta 9 = 0
% H1; there is heteroskedasticity present
RSS = (residuals.'*residuals);
sigmaeps = (1./T).*RSS;
epsnew = (residuals.^2)./sigmaeps - 1;
Beta_BP = inv(X.'*X)*X.'*epsnew;
epsfitted = X*Beta_BP;
BP_stat = sum(epsfitted.^2)./2
bpcrit = chi2inv(0.95,m)
pvalue_BP = 1-chis_prb(BP_stat,m)
```

Fig(2.14)

We see that since we are approximating the test statistic rather than using the 'F-statistic', it becomes a χ^2 distribution. The code provides the following results:

BP	BP Critical Value	P-Value
1195.8	15.5073	0

Therefore, we can clearly see that we *reject* the $\mathbf{H_0}$ at the 5% significance level – and in fact at all significance levels since the **p-value** is 0. We can conclude that there is **heteroskedasticity** that is dependent on the regressors (*i.e.* **proportional heteroskedasticity**).

Possible fixes for this would be to either use:

- The "Generalised Least Square Estimator" which would involve a transformation but would result in restoring the efficiency of the estimated coefficients.
- Additionally, we could improve the *estimation* of the *standard error* of the regression by taking a robust variance-covariance matrix – however this would involve the use of the "**White Robust Standard Error Estimator**".
- Bootstrapping or clustering is also a possibility.

(iii) Serial Correlation

The test for **serial correlation** is the "**Durbin-Watson Test**" and in this test we assume that errors (residuals) are represented by an **autoregressive model** of **order 1** [**AR(1)**]. The AR(1) model can be written as follows:

$$\varepsilon_t = \rho\varepsilon_{t-1} + \upsilon_t$$

In this function, ρ is the **autocorrelation coefficient** and it is called '*autocorrelation*' because it is essentially the correlation of the error term with itself. Under H_0 we have $\rho = 0$ which means *no serial correlation*.

There is a rule of thumb which we employ for this test, using the **Durbin-Watson test statistic** (**d**):

- If **d** = 2 then $\rho = 0$ [no serial correlation].
- If **d** = 0 then $\rho = 1$ [positive serial correlation].
- If **d** = 4 then $\rho = -1$ [negative serial correlation].

We calculate '**d**' using the following code:

```
%% 9. Residuals Diagnostic Test for Serial Correlation [Part 1 (g)]
```

```
% Durbin-Watson test
% H0: rho = 0
% H1: eps_{t} = rho*eps_{t-1} + u_{t}, with rho different from 0.
DW=sum(diff(residuals,1).^2)./sum(residuals.^2)
```

Fig(2.15)

This will give us our test statistic which is as follows:

D-W Statistic
1.9778

The result is that **'d'** ≈ 2 but this is not exact as it involves rounding up. Therefore, it is necessary to construct an interval to determine what type of **serial correlation** exists if any – or there could be the scenario where we have an '**inconclusive result**' where we cannot determine for certain whether *serial correlation* exists or not.

The interval is cumbersome to explain but intuitive to read, therefore this table will help explain:

0	dL = 1.917	dU = 1.936	2	4 - dU = 2.064	4 - dL = 2.083	4
Positive SC	Inconclusive Area		No SC	Inconclusive Area		Negative SC

- '**dL**' represents the *lower bound* of the **critical value** of the **DW statistic**.
- '**dU**' represents the *upper bound* of the **critical value** of the **DW statistic**.
- The period between '**dU**' and **'4 - dU**' represents the range within which we can conclusively say that there is **no serial correlation**.
- Our value of **1.9778** falls within this range, as it is close to 2 and higher than the *upper bound*.

Conclusively, we can say that we fail to reject H_0 meaning that there is **no serial correlation** amongst the **residuals** of the fitted model and that $\rho = 0$. This is a welcome result after finding a significant degree of proportional heteroskedasticity amongst the residuals.

h) Multicollinearity

The final piece of our OLS regression function is to test for **multicollinearity** – this occurs when *2 or more* of our regressors are **correlated**. Even if our model runs perfectly and looks good, there can still exist multicollinearity so it is always necessary to test for this.

For the final part of the report, the '**Variance Inflation Factors**' (**VIFs**) of the regressors will be calculated. This is done by ***regressing*** each **regressor** onto each other – essentially performing numerous **auxiliary regressions** with each regressor as a dependent variable. However, MatLab provides a much easier way of doing this through constructing a correlation coefficient matrix of the regressors:

```
%% 10. Testing for Multicollinearity [Part 1 (h)]
R0 = corrcoef(X(:,2:end));
VIF = diag(inv(R0))';
VIF = array2table(VIF)
```

Fig(2.16)

[Note; we are disregarding the constant as this will return an *irrational* **VIF** value and it is not a regressor]

The **VIFs** can be calculated by taking the **transpose** of the ***main diagonal*** of the **inverse** of the **correlation coefficient matrix**. This is because the formula for **VIF** is:

$$VIF = \frac{1}{1 - R^2}$$

From this equation we can see that if the *auxiliary regression* has **high correlation** (a high **coefficient of determination**) then the **VIF value** will be high. In general:

- A **VIF** value less than 5 can be seen as having *no multicollinearity* or *minimal collinearity*.
- A **VIF** value between 5–10 means there is multicollinearity but it is not problematic.
- Anything greater than 10 means that multicollinearity is an issue.

Running this code will return a table for the **VIFs** as follows:

VIF1 (X2)	VIF2 (X3)	VIF3 (X4)	VIF4 (X5)	VIF5 (X6)	VIF6 (X7)	VIF7 (X8)	VIF8 (X9)

1.6031	3.0252	4.8419	1.8365	1.5134	1.0858	1.3118	3.0561

All of the **VIF** values are *less than* 5 which means that there exists either **no multicollinearity** or **minimal multicollinearity** between our regressors. This means no further steps, such as employing Machine Learning, are required as our regressors are not correlated.

Conclusion

The OLS Regression Model that has been constructed is a well-fitted model but still has room for improvement and would no doubt benefit from the addition of additional regressors to further improve the **'goodness of fit**'. Interesting conclusions have been drawn from the estimated coefficients, despite 2 of them being statistically insignificant. Removing these regressors and adding new ones could help improve the model greatly, but would require further research and data collection.

The residuals exhibit a **leptokurtic distribution** and as such, transforming some of the regressors using *natural logarithmic transformation* could help aid this issue or using robust regression. We could also conclude that there is no longer a normal distribution and substitute a '***t-distribution***' into the model. However, it is likely that the distribution is not normal due to **heteroskedasticity** and instead the recommendation from this report is to focus on solving that issue.

The model has **proportional heteroskedasticity** and this is an issue that needs to be solved – the best course of action would be to use the "**White Robust Standard Error Estimator**" or any similar numerical algorithm in order to calculate a more *robust standard error* (whilst maintaining the same Betas). Fixing this could resolve many of the issues that the model currently has.

There appears to be *no serial correlation* in the residuals – however, this could change with the addition of new regressors. In a similar fashion, there is no issue of *multicollinearity* but if changes are made to the model then these conclusions may

not hold. Therefore, changes should be made with caution and instead the focus should be on eliminating the heteroskedasticity present in the model.

UCD Michael Smurfit Graduate Business School

Part 3

Introduction

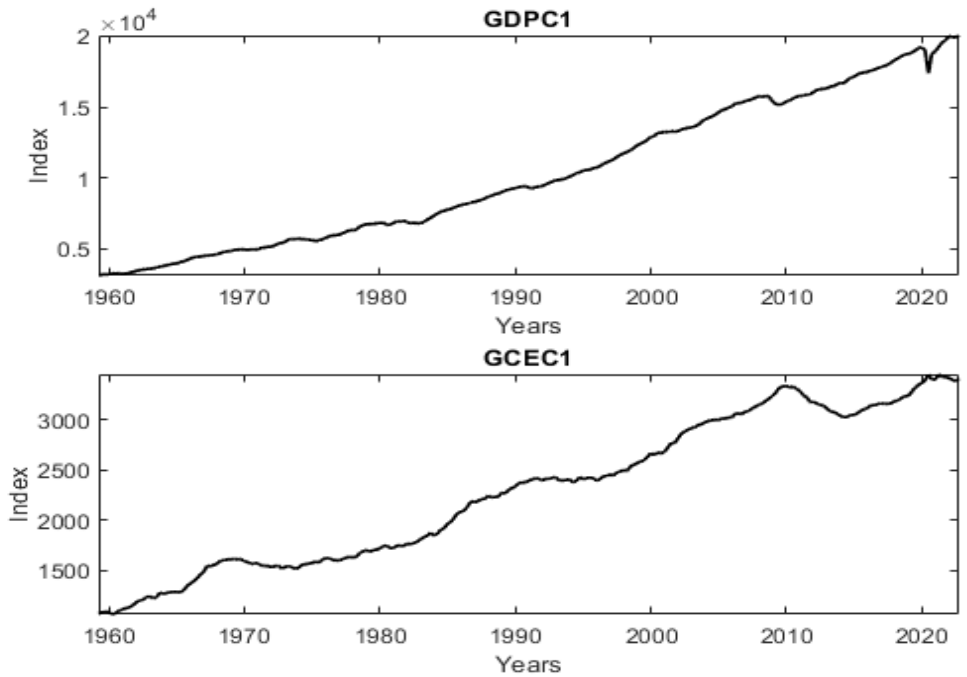
This section of the report will deal with ARIMA forecasting – where one time-series was extracted from the website of the Federal Reserve of St. Louis and provided by Michael W. McCracken who has it possible to access macroeconomic data that is automatically updated with each time period.

The time-series in question is the GDP of the United States of America (USA) which is provided on a quarterly basis. This time-series will be analysed closely before forecasting using an ARIMA Model. Note, the code used to produce each of the displayed outputs is not included in this part of the report as it would become too long. However, it will be available in the attached 'm' file.

a) Identifying Trend

Plot of the Time Series

As mentioned, the data of interest is the quarterly GDP (GDPC1) of the US economy. This data was collected from the FRED-MD website and plotted in order to see if any information could be gleaned from a plot of the raw data. In addition, a subplot was created to show the time-series of Real Government Expenditure (GCEC1) in order to determine if these time-series exhibit similar or different characteristics to draw conclusions from:

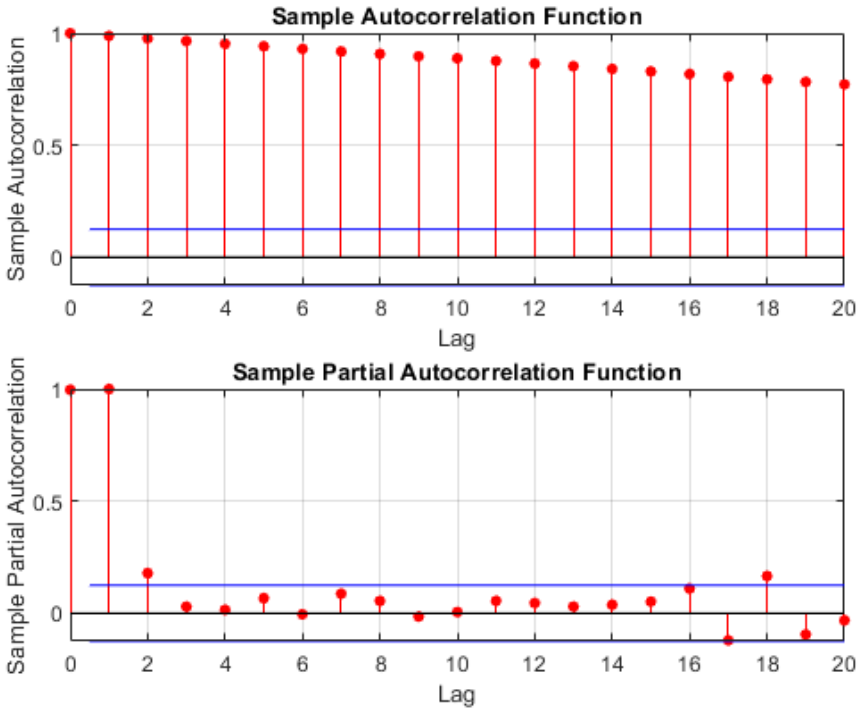


- Looking at the above graph we can see that the process is not mean-reverting since we assume the mean is equal to 0 and typically a **'stationary process'** will revert to the mean with lags – considering an **Autoregressive model of Order One** [**AR(1)**], we would expect that it would revert to the mean. However, the opposite occurs and therefore it is likely that this is a **non-stationary process.**
- The plot of the time-series seems to suggest that there is a **stochastic trend** present which means that we have to consider that this is likely a **random walk.**
- A random walk describes a non-mean-reverting process and is also referred to as an **Integrated Process of Order One** – it is likely that this is an **I(1)** as there is a **unit-root** present.
- We can conclude this because in the **AR(1)** we would expect the **autoregressive coefficient** ϕ to be *less than* 1 if the process is **stationary**, which we know it is not**.** Conversely, a non-stationary process exhibits the characteristic of a **unit-root** which describes the scenario where '**ϕ = 1**'.

However, to be thorough we will consider the autocorrelation

Correlogram of the Time-Series

It is necessary to consider a correlogram of the time-series:



- Looking at the above correlogram we can see that this model exhibits a '**trend in variance**' which is another way of saying *random walk*, **I(1)** and *unit-root*. This is due to the '**persistence**' of the **empirical autocorrelation function**.
- If 'ϕ = 1' then, as has been discussed, we have a **stochastic trend** or a **trend in variance**. The **empirical autocorrelation function** will show this through gradually decreasing spikes outside the normal interval (blue line). This is because the autocorrelation coefficient is calculated at each lag by taking the following exponential of the **autoregressive coefficient**. Giving us the general formula:

  $\varrho = \phi^{k}$ [where k = no. of lags]

  Therefore, since the **autoregressive coefficient** is equal to 1 it will only slowly decrease. This behaviour is characteristic of a **trend in variance**.

However, there could be the case that there is a **trend in mean** and a **trend in variance** – the **autocorrelation function** is the exact same for this case as it is for the **stochastic trend**. Therefore, we move on to our last step in the process of identifying the correct trend.

Augmented Dickey-Fuller Test

This test is necessary in order for us to determine whether we have just a **trend in variance** or both a **trend in variance** & **trend in mean**. The latter is referred to as a **random walk** with **drift**. It includes a **'constant'** term.

Under the *null hypothesis* ($\mathbf{H_0}$) of the **Dickey-Fuller Test** we assume that $\gamma = 0$ where '$\gamma = (\phi - 1)$'. Therefore, if we *fail to reject* $\mathbf{H_0}$ then we assume a **unit-root** because gamma will only equal 0 when the **autoregressive coefficient** equals 1 which will mean we have an **integrated process** / **random walk**.

Firstly, we confirm that we in fact have a **trend in variance** by running the **Augmented Dickey Fuller Test** with no deterministic trend (a deterministic trend

is a trend in mean). Running the test in MatLab provides us with the following output:

```
 T-stat     Pval (Lag = Auto (BIC), No Deterministic)
 7.3404     1.0000
```

A **p-value** of 1 means we *fail to reject* $\mathbf{H_0}$ and implies a **unit root**.

Secondly, we need to determine if we have a **random walk** with **drift** which involves the inclusion of a **constant term** in our auxiliary regression. Running the code in MatLab provides the following results:

```
T-stat     Pval (Lag = Auto (BIC), Intercept)
1.8695     1.0000
```

A **p-value** of 1 means we *fail to reject* $\mathbf{H_0}$ and implies that we have a **random walk (unit-root)** with **drift**.

Finally, we will check to see if we have a linear-trend and a drift (intercept) in order to determine if this is a **random walk** with **drift** (a **trend** in *both* mean & variance). Running the **Augmented Dickey Fuller Test** in MatLab provides us with the following result:

```
  T-stat     Pval (Lag = Auto (BIC), Intercept & Trend)
 -1.8043     0.7060
```

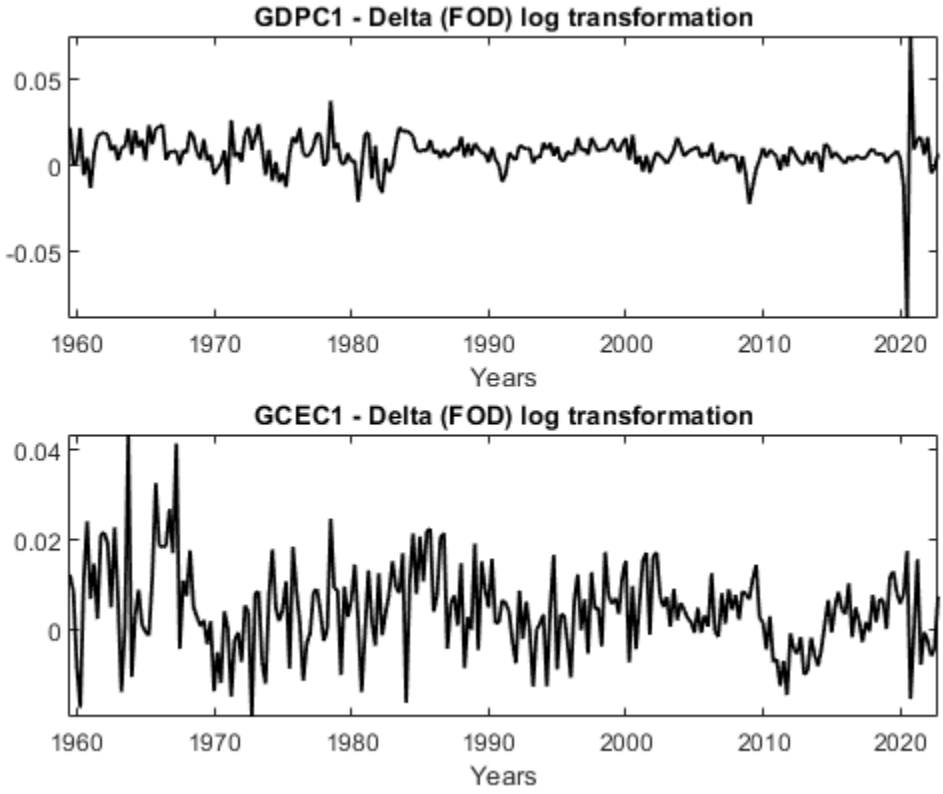
This result shows us that we do in fact have a **random walk** with a **linear trend** and **drift**. This means we have a **constant**, a **linear trend** and a **unit-root**.

b) Data Transformation

The preceding step was necessary in order to identify what type of trend is present in our data. It is now necessary to remove these trends in order to achieve a **stationary process**. A benefit of using the FRED-MD website is that the data comes complete with a '**T-code**' for transformations to make the time-series stationary. Rather than doing this manually we can apply the **T-code** to our **time-series** and (hopefully) achieve a **stationary process** that we can begin to analyse in order to predict what model should be used before creating ARIMA Models to use for forecasting.

Plot of the Transformed Time Series

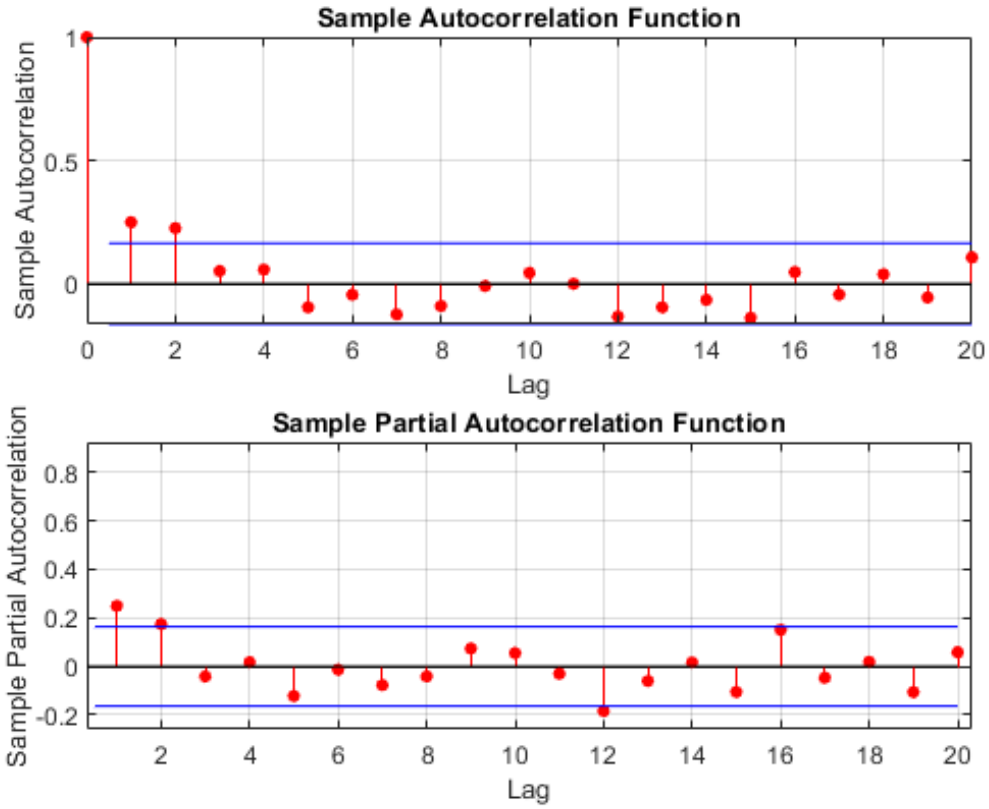
The transformation will again be applied to GDP and Real Government Expenditure to see how both time-series will change when subject to the same transformation. According to FRED-MD they should be transformed using the same **T-code (5)** so we can compare these 2 time-series to determine whether they exhibit the same characteristics when compared with the plot of the raw time-series data:



- Looking at this post-transformation model, we see that we have in fact created a time-series that looks like it is '**stationary**'.
- It appears to be **mean-reverting** with **lags** and this is characteristic of a **stationary series**.

Correlogram of the Time Series

It is now necessary to consider the correlogram of the transformed **time-series** to see if we can begin to guess what type of model fits our data:



- The empirical autocorrelation function rapidly approaches 0, it reaches it after the third lag which is characteristic of an **Autoregressive Model (AR)**.
- When considering an **AR Model** it is important to consider the **partial autocorrelation function (PACF)** to determine the **number of lags**.
- In the **PACF** we see that there is a *strange pattern* being shown where the spikes are alternating and exceeding the normal interval sporadically.
- This strange pattern suggests that we have an **ARMA Model** – however it is impossible to determine from just visual inspection of a plot and correlogram what number of lags we have in the **autoregressive** and **moving average** components.

c) ARIMA Model Estimation

Log-Likelihood Estimation & Schwarz Information Criterion

Having identified that it is likely we are dealing with an **ARMA Model** it is necessary to estimate a number of **ARMA models** with *different lags* in order to determine which best suits our **time-series** for *forecasting purposes*.

Using log-likelihood, we estimate an array of ARMA Models ranging from **ARMA(1,1)** to **ARMA(4,4)** and compare the models using the **Schwarz Information Criterion (BIC)**. This was all done in MatLab and produced the following results where the **BIC** is presented in an array as such:

	MA(1)	MA(2)	MA(3)	MA(4)
AR(1)	-974.502	-971.329	-965.5506	-965.5475
AR(2)	-972.338	-967.8221	-960.9454	-980.3883
AR(3)	-967.9933	-970.5086	-956.3402	-958.0123
AR(4)	-963.3984	-966.1865	-957.8674	-972.576

This table produces the **BIC** for each combination of **ARMA(p,q)** where p and q represent the number of lags in the **AR** and **MA** components of the model respectively.

When considering **information criteria** the *lower* the value the better – when considering negative values this means the one with the **highest modulus** (*i.e.* the number *deepest* in the negative). From the table we can conclude that the best model to use is **ARMA(2,4)**. Therefore, we will take this model and check it's estimation to ensure that all the coefficients (*i.e.* ϕ and θ) are **statistically significant:**

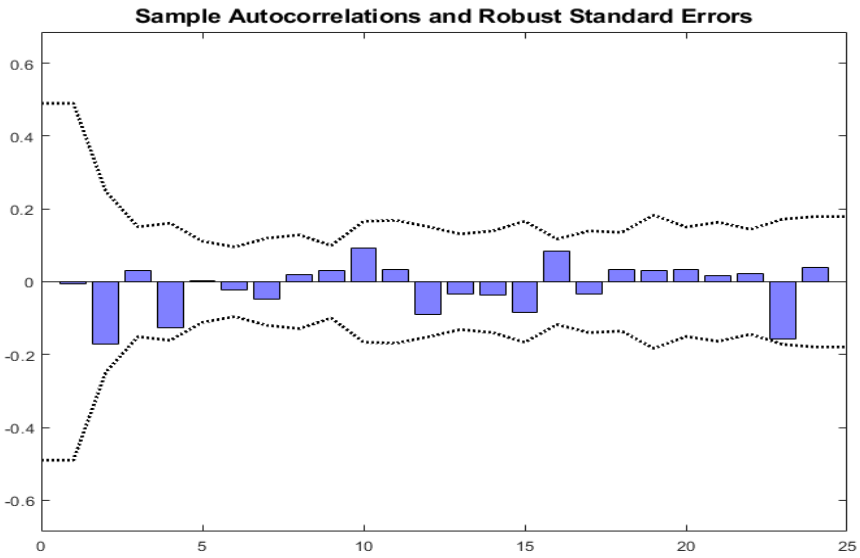
ARIMA(2,0,4) Model (Gaussian Distribution):

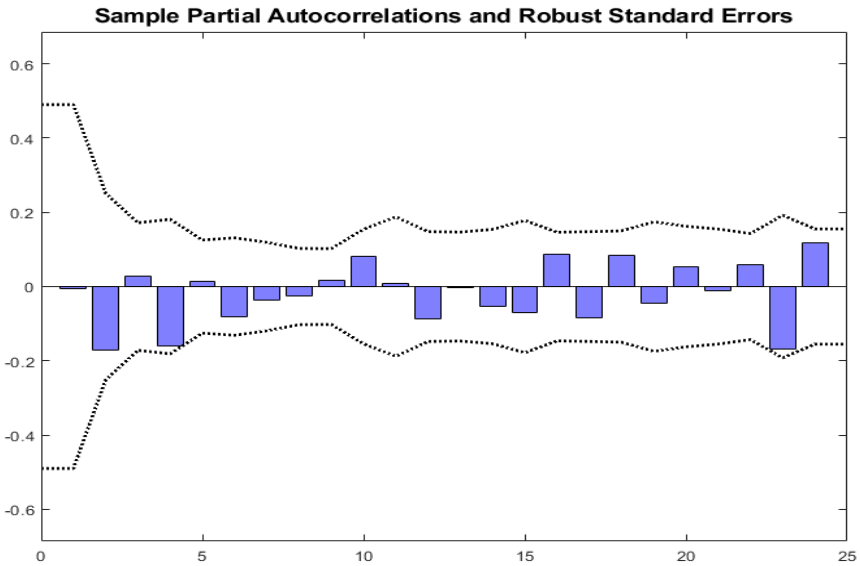
	Value	StandardError	TStatistic	PValue
Constant	0.026557	0.0033313	7.9719	1.5543e-15
AR{1}	-1.4123	0.031041	-45.497	0
AR{2}	-0.75921	0.030048	-25.266	0
MA{1}	1.7286	0.11286	15.317	0
MA{2}	1.4133	0.20519	6.8877	5.6704e-12
MA{3}	0.48717	0.17908	2.7205	0.006519
MA{4}	0.232	0.07875	2.946	0.0032195
Variance	6.5458e-05	6.8913e-06	9.4986	0

Looking at the output from our **ARMA Model Estimation** done in MatLab we can deduct from the **p-values** that all of our **estimated coefficients** are **statistically significant** for *any significance level* (alpha = 1%, 5% or 10%).

Residuals Diagnostics

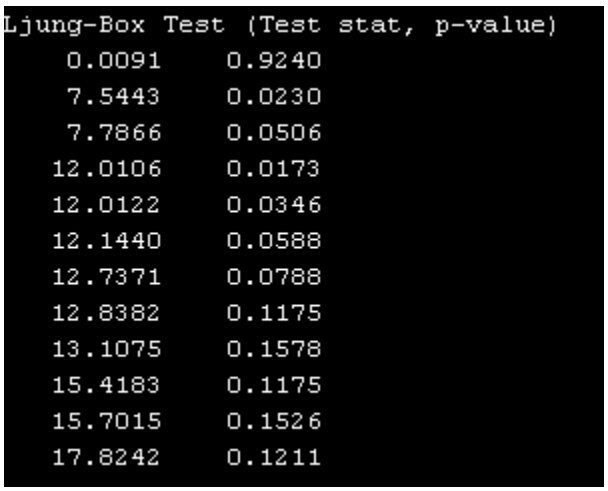
Whilst the **ARMA(2,4) Model** seems to fit well, we need to consider whether there is **serial correlation** in the **residuals** which is done using the "armaxfilter" command in MatLab that is very useful for creating the desired ARIMA and GARCH models. Running this code and plotting the *standardised errors* (used because they are scale free) the following 'Autocorrelation' and 'Partial Autocorrelation Function Plots' were produced:





- From these plots we can see there should be very little issue with *serial correlation* in the residuals. It seems that there is *no serial correlation present*.

In order to definitively determine if there is the presence of autocorrelation we can implement the "**Ljung-Box Test**" which has as its null hypothesis ($\mathbf{H_0}$) that there is *no serial correlation*. Running the necessary code in MatLab produces the following results:

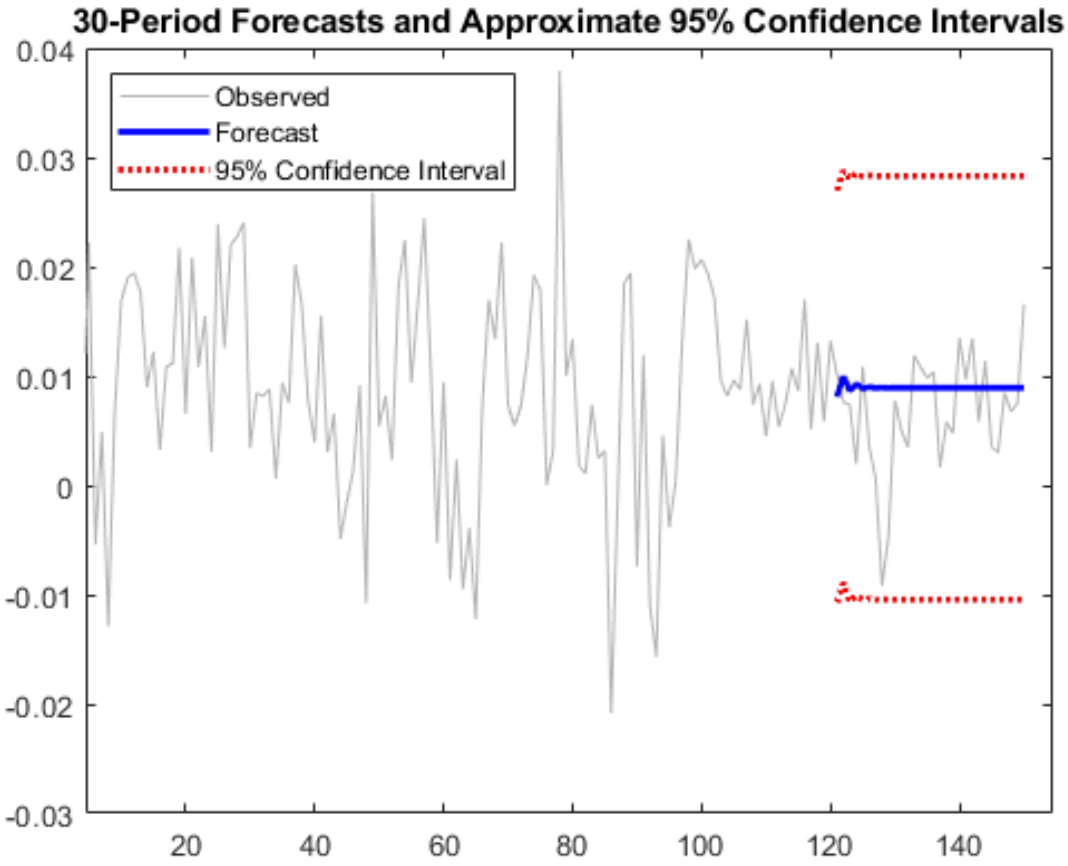


Therefore, since the P-value are high across the board we can conclude that at the 1% significance level there is **no serial correlation** amongst the residuals as we *fail to reject* $\mathbf{H_0}$ at this significance level.

d) Forecasting 'Pseudo' Out-of-Sample

ARMA(2,4) Forecast

With our decision made on using the **ARMA(3,3) model** after confirming that it is suitable for use, we can begin to use it for forecasting purposes. The time period we will be using are the first 150 periods (quarters) and we will be forecasting from the 120th quarter to the 150th quarter meaning our forecast horizon is 30 periods. Firstly, we will forecast **pseudo out-of-sample** which means we will be able to calculate our forecast error as we will have the observed value of the time-series for the horizon over which we are forecasting. This will allow us to then calculate the "**Root Mean Square Forecast Error"** (**RMSE**) which we can then use to compare our model against other models in a manner different from using **information criteria**. Running the necessary code in MatLab produced the following forecast:



We can see that the forecast is in blue and it lies within the 95% confidence interval. Using this forecast, we can calculate the **forecast error** which is the difference between the **observed value** of the time-series (the *light grey line*) and the **forecasted value** of the time-series (the *blue line*). Calculating this we can then get our RMSE and after running the necessary code in MatLab we produce the following result:

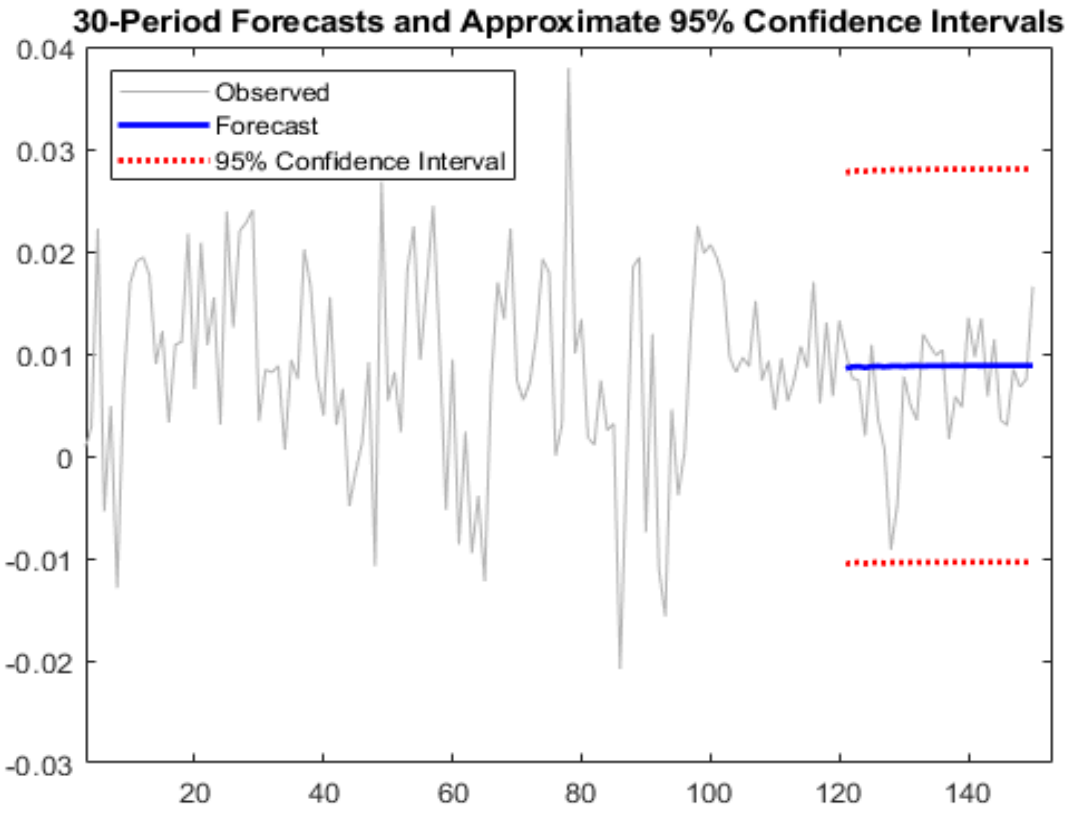
RMSFE (M1)
3.353E-05

This our *root mean square forecast error* for our first model **ARMA(2,4)**. This is necessary in order to compare with another model of our choice.

ARMA(3,3) Forecast

For our second model we will choose **ARMA(3,3)** in order to compare the forecasts and to determine if the difference in forecast errors is statistically significant ("**Diebold-Mariano Test**").

Running the same code in MatLab but altering the *autoregressive lags* to 3 and the *moving average lags* to 3 - we produce the following plotted forecast:



It is similar to our previous forecast of course since it used the same horizon but we can see a difference in the forecasted line (*blue*) as it has a different shape which means a different forecast. However, the shape of the line is not important for comparing these models. It is necessary to calculate the RMSE of the second model and compare them with each other which was done in MatLab producing:

```
RMSE =

   1.0e-04 *

    0.3353    0.3282
```

This is a very peculiar result and not expected whatsoever. Our RMSE for the **ARMA(3,3)** is actually lower despite the **ARMA(2,4)** having the *lowest **BIC*** value! However, this is not damning if we find that the difference between them is actually statistically insignificant.

The 'Diebold-Mariano Test'

This test has under its null hypothesis ($\mathbf{H_0}$) that the Mean Forecast Error of the **ARMA(2,4)** is equal to the Mean Forecast Error of the **ARMA(3,3)**:

$\mathbf{H_0}$**;** EFm1 = EFm2

We run this test in MatLab and the following result is produced:

```
DMstat =

    1.3880


pvalue =

    0.1757
```

Thankfully we can conclude from this that the **Forecast Errors** are **statistically the same** at all levels of significance (1%, 5% and 10%).

Conclusion

After carrying out pseudo forecasts of 2 models we have reached the conclusion that either of these 2 models are suitable for use despite **ARMA(2,4)** having the *lower* ***Bayesian Information Criterion***. Whilst this is a strange result it is not uncommon, particularly given that our series is not extremely volatile as shown by the **Ljung-Box test**. With this in mind, we still stick to our guns and continue to use the **ARMA(2,4) Model** for the last part of this report.

e) Forecasting 'Out-of-Sample'

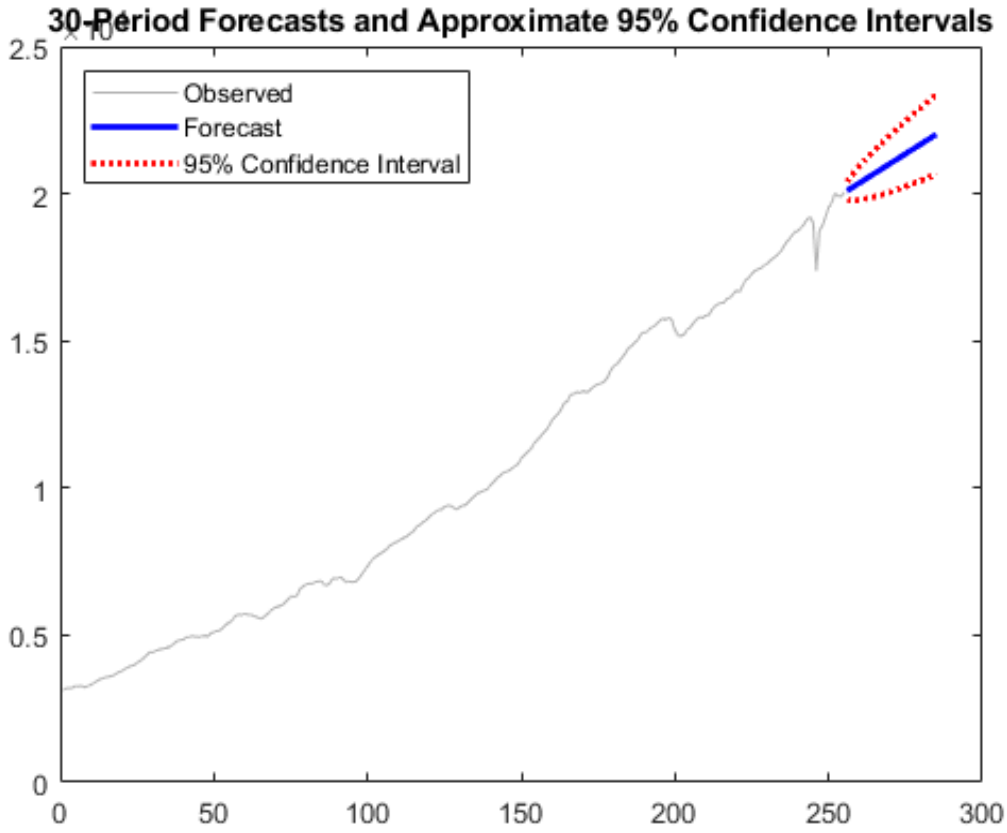
The 'Naïve Model'

The final part of this report will focus on using our estimated model to perform an actual '*out-of-sample*' forecast that cannot be compared with other models until the necessary data for the next horizon period (30) becomes available. We will not be able to calculate our forecast errors if we do not know the actual observed value!

For our 'out-of-period' forecast we are going to consider the '**Naïve Model**' which is making a forecast with a random walk. As we already discovered, our time-series includes a **random walk** with **drift** and a **linear trend**. This is because in a **random walk,** if there are no shocks, then it is likely that there will not be a big increase or decrease in GDP. Given the number of shocks the world has had to endure recently, I hope that there will be no more in the immediate future!

The **Naïve Model** can be seen as an excellent forecaster in these circumstances and can be used as a **comparison** for **more complex models**. Therefore, in the future when the relevant data becomes available we can calculate the **forecast error** of our forecast with the **'out-of-sample' random walk** with our **ARMA(2,4)** in order to determine if our more complex **ARMA(2,4)** is truly a good model to use for forecasting purposes. Note that the forecast is carried out on our original non-transformed data as necessitated by it being done as an **integrated process**.

Running the necessary code in MatLab provides us with the following forecast:



Following the trend that we have seen and identified, the forecast predicts that this trend will continue almost linearly as is expected from the **Naïve Model** as it is based on using the last period of the 'h' horizon and assuming no big shocks occur. Only time will tell if our **ARMA(2,4) Model** will be adequate as we compare the errors between it and the **Naïve Model.**

f) Conclusion

As we have seen, the **ARMA(2,4) Model** is not necessarily the only model that is perfectly suitable for forecasting **GDP** of the US Economy. We also have the **ARMA(1,1) Model** which provides forecast errors that are statistically the same as

our proposed model. Forecasting with the **Naïve Model** will allow us to compare these models in the future and hopefully provide a more concrete result that allows us to identify which of these models, if any, are most suitable going forward.