TASK 2

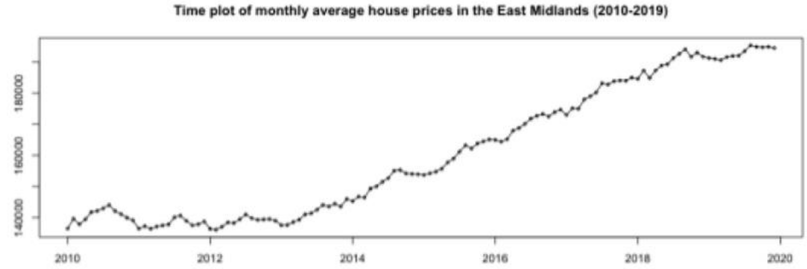
East Midlands House Price Analysis and Future Forecast Report

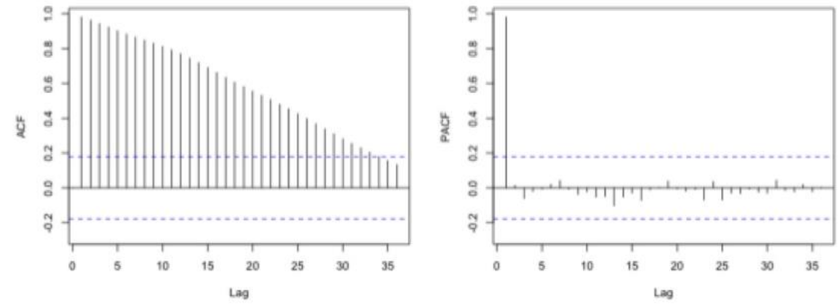
Executive Summary

This report focuses on analyse a dataset of monthly average house prices in the East Midlands form 2010 to 2019. This report is a reference for the local government agency. As a result, we will forecast monthly house prices for the first six months of 2020 using the time series. After several model tests we chose the ARIMA(1, 1, 2) × (0, 1, 1) [12] model, which predicts that house prices will still show an upward trend in the first six months of 2020.

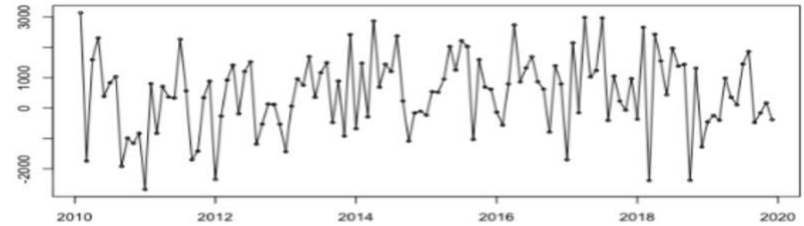
I. Introduction

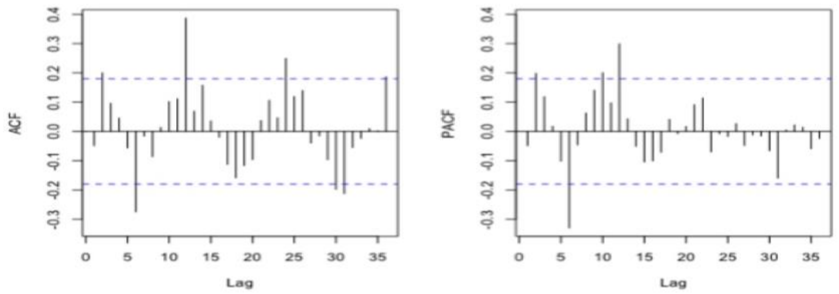
Taking a look at the overall data, and setting the monthly average prices as time series data, you can see that the years 2010-2013 have seasonal character, with house prices showing a general upward trend after 2013. In 2019, a declining inflection point was followed by a rebound. It is clear from the autocorrelation function (ACF) plot that the data are extremely volatile (very slow decay).



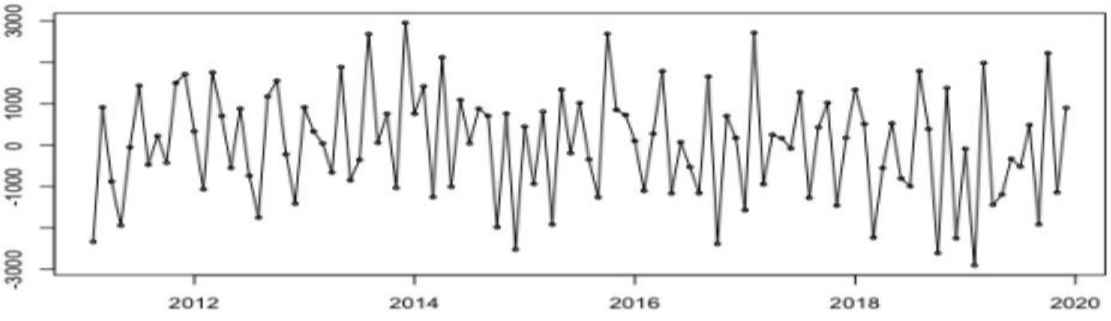


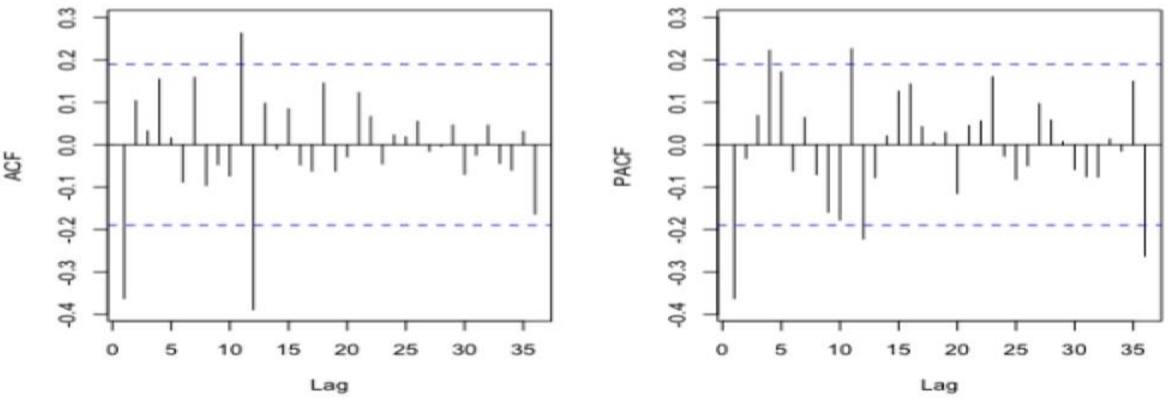
First we need to convert the data to stationary data. When we look at the graph with one lag using the difference method, the graph looks basically oscillating around 0 and is basically flat. However, the ACF graph shows that there is a 12th order seasonal influence, with spikes at 12, 24, 36.





So we difference seasonally at lag 12.From the time plot of the seasonally differenced data shown below.





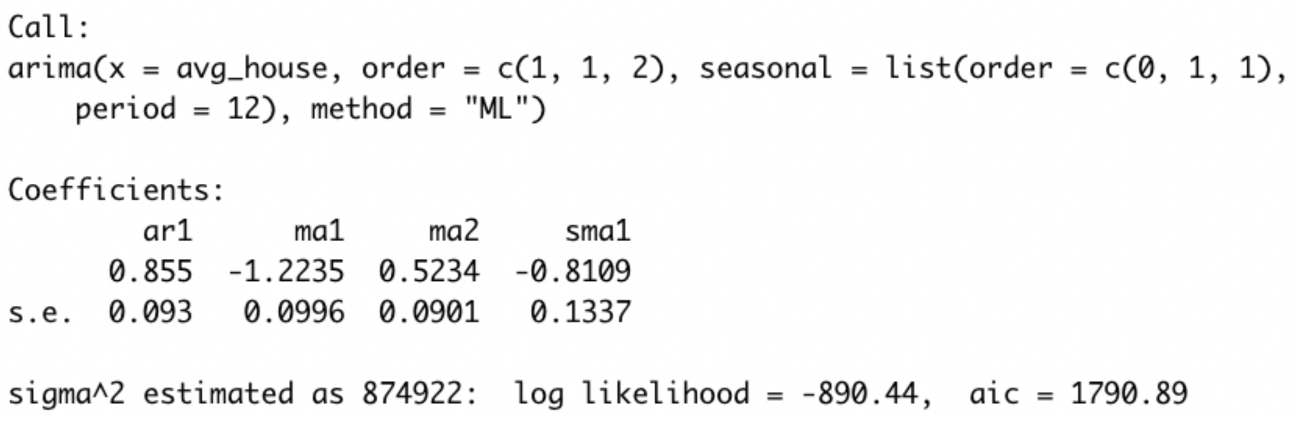
The seasonality appears to have been removed, and these data certainly appear to be more stationary (the mean appears to be constant and equal to zero).

At this point we'd assume that these differenced data are stationary. So it will be sugggest that we'll fit an ARIMA(p, 1, q) × (P, 1, Q)[12] model to the original data. For the non-seasonal component an ARMA (1, 1) model can be used first.

In the next step, we fitted the models to see which one was best, and the results are shown in the following table:

Model	AIC	L-jung test P-value of residuals	Fit or not
ARIMA(1, 1, 1) × (0, 1, 0) [12]	1831.9	P<0.05	Not fit
ARIMA(1, 1, 1) × (0, 1, 1) [12]	1808.48	P<0.05	Not fit
ARIMA(1, 1, 1) × (1, 1, 0) [12]	1816.82	P<0.05(lag>4)	Not fit
ARIMA(1, 1, 1) × (1, 1, 1) [12]	1809.17	P<0.05(lag>4)	Not fit
ARIMA(1, 1, 2) × (0, 1, 0) [12]	1825.04	P>0.05	Fit good
ARIMA(1, 1, 2) × (0, 1, 1) [12]	1790.89	P>0.05	Fit good
ARIMA(1, 1, 2) × (1, 1, 0) [12]	1810.82	P>0.05	Fit good
ARIMA(1, 1, 2) × (1, 1, 1) [12]	1792.74	P>0.05	Fit good
ARIMA(2, 1, 1) × (0, 1, 0) [12]	1833.84	P<0.05(lag=4)	Not fit
ARIMA(2, 1, 1) × (1, 1, 0) [12]	1812.98	P>0.05	Fit good
ARIMA(2, 1, 1) × (0, 1, 1) [12]	1795.24	P>0.05	Fit good
ARIMA(2, 1, 1) × (1, 1, 1) [12]	1797.16	P>0.05	Fit good

According to the less AIC value and the number of parameters, the better model fit. The following is the ARIMA(1, 1, 2) × (0, 1, 1) [12] fitted model output:

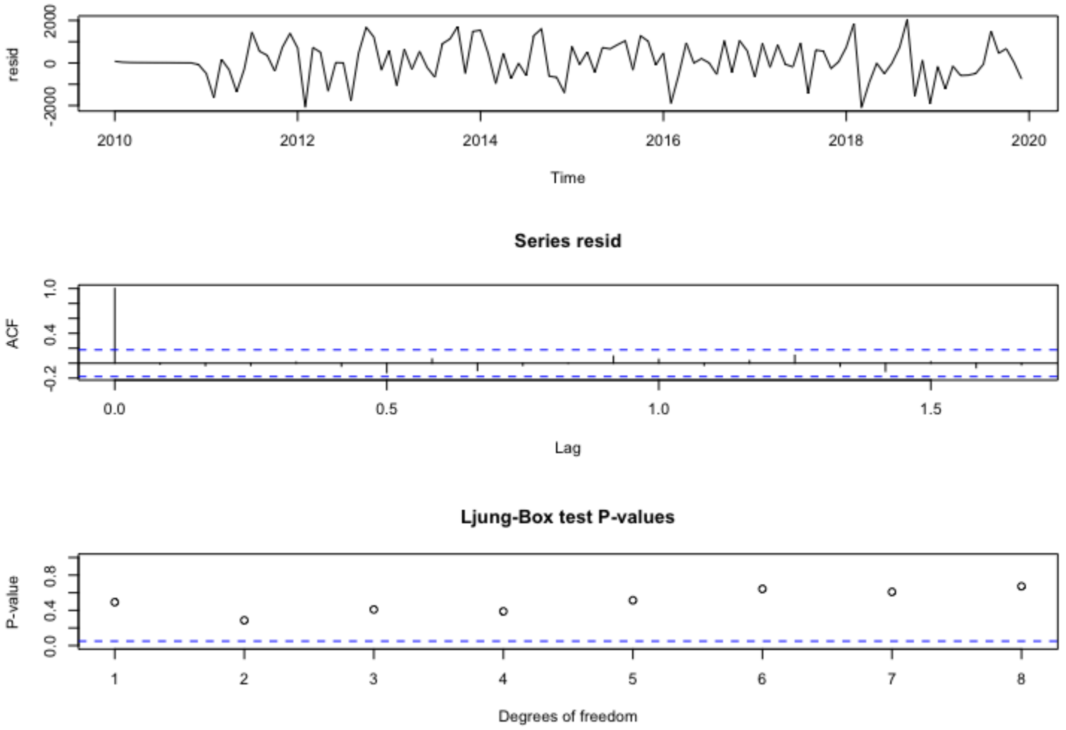


```
Call:
arima(x = avg_house, order = c(1, 1, 2), seasonal = list(order = c(0, 1, 1),
    period = 12), method = "ML")

Coefficients:
         ar1      ma1     ma2     sma1
       0.855  -1.2235  0.5234  -0.8109
s.e.   0.093   0.0996  0.0901   0.1337

sigma^2 estimated as 874922:  log likelihood = -890.44,  aic = 1790.89
```

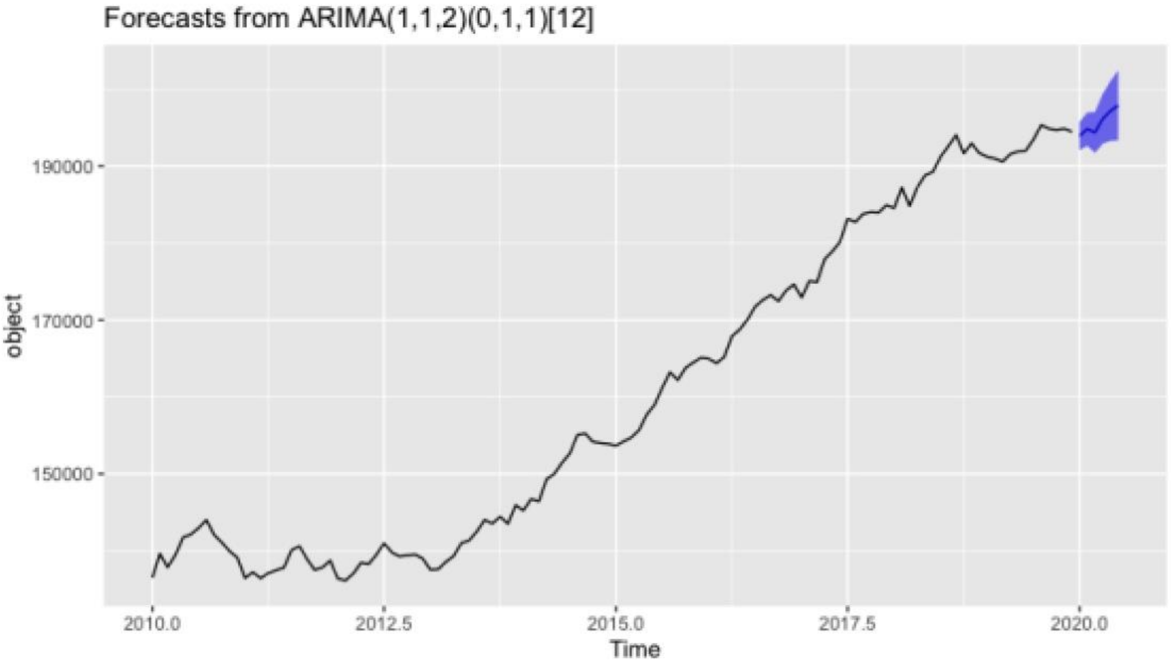
For the residuals, we produce a time plot, plot of the sample ACF against the lag and a plot of the Ljung-Box test statistics, shown below:



Finally we choose ARIMA(1, 1, 2) × (0, 1, 1) [12] could be a reasonably good fit. The residuals look like white noise (no correlation between residuals) and the Ljung-Box test statistics are all fairly large.

II. Conclusion

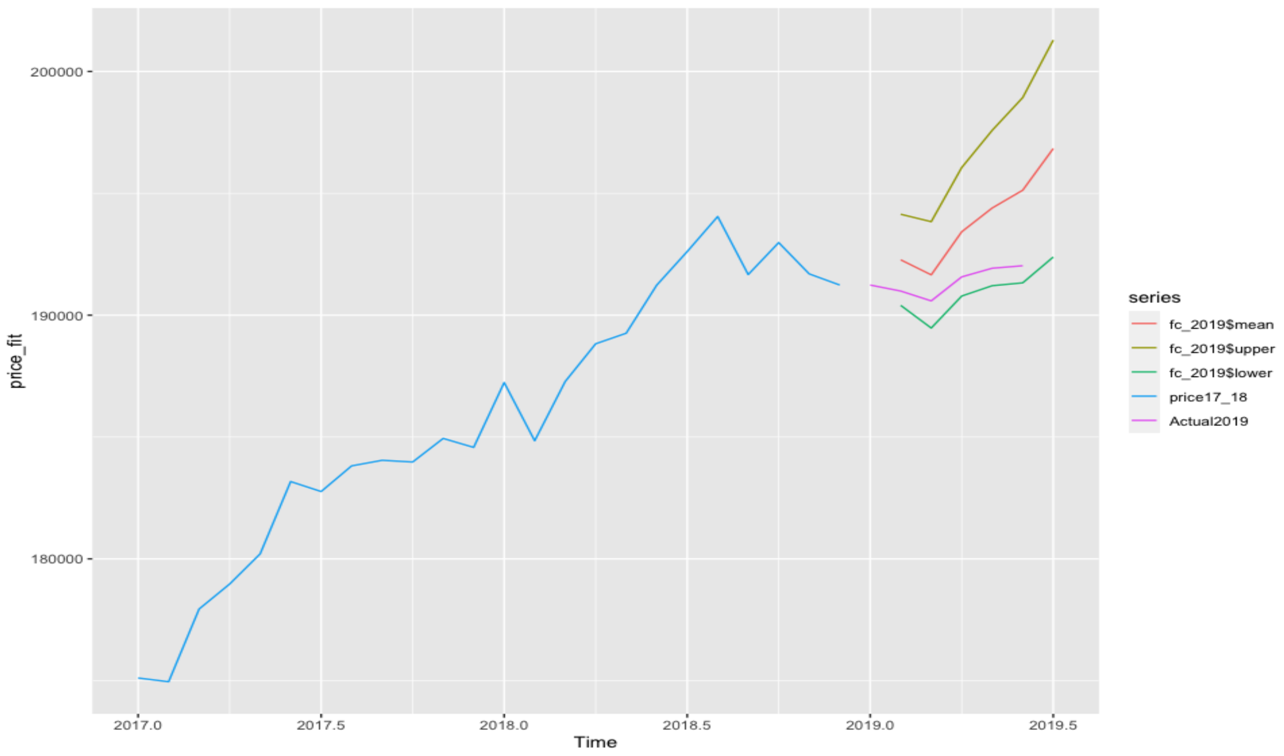
We use the above model to forecast house prices for the next six months in 2020, as shown in the chart below：



In the graph you can see that house prices will still rise in the future, as indicated by the blue forecast data. So, the house price in 2020 are predicted as following table:

Year	**Month**	**Forecast price**	**Lowest price in 95% confidence interval**	**Highst price in 95% confidence interval**
2020	Jan	193930.5	192090.5	195770.5
2020	Feb	194836.4	192660.8	197012.0
2020	Mar	194401.0	191733.3	197068.7
2020	Apr	196204.1	192945.6	199462.5
2020	May	197202.1	193297.8	201106.3
2020	Jun	197933.9	193356.4	202511.4

Now we'd like to see how our forecasted values compare to the observed values. In order to fit the model, we take the values of the series in 2017 and 2018, then extrapolate the first six months of 2019 house prices. To do this, we produce a plot that shows both the true values and the forecasted values.



The graph above shows that our model predicts prices that are closer to the true values. If the sample size were larger, the model would fit more accurately.

Appendix

I. TASK 1 R CODE

```
#read the data
cet_temp <- read_csv("Desktop/20353100-MATH4022-CW/cet_temp.csv")
#annual mean temperature set as time series data
annual_mtemp<-ts(cet_temp$avg_annual_temp_C,start=1900,frequency=1)
par(mfrow=c(3,1))
ts.plot(annual_mtemp,ylab="Annual mean temperature",xlab="Year",
      main="Time plot of annual mean temperature in the Midlands region of England (1900-
2021)")
#Plot the sample ACF
acf(annual_mtemp)
#Plot the sample PACF
pacf(annual_mtemp,ylim=c(-1,1))


annual_mtemp2<-diff(annual_mtemp)
par(mfrow=c(3,1))
ts.plot(cet_temp2)
#Plot the sample ACF
acf(annual_mtemp2,main="Series annual_mtemp2")
#Plot the sample PACF
pacf(annual_mtemp2,ylim=c(-1,1),main="Series annual_mtemp2")


# AR(1) model
model.AR1<-arima(annual_mtemp2,order=c(1,0,0),method="ML")
arima(x = annual_mtemp2, order = c(1, 0, 0), method = "ML")
resid.AR1<-residuals(model.AR1)
par(mfrow=c(3,1))
ts.plot(resid.AR1)
acf(resid.AR1,main="Series resid.AR1")
```

```
LB_test<-function(resid,max.k,p,q){
  lb_result<-list()
  df<-list()
  p_value<-list()
  for(i in (p+q+1):max.k){
    lb_result[[i]]<-Box.test(resid,lag=i,type=c("Ljung-Box"),fitdf=(p+q))
    df[[i]]<-lb_result[[i]]$parameter
    p_value[[i]]<-lb_result[[i]]$p.value
  }
  df<-as.vector(unlist(df))
  p_value<-as.vector(unlist(p_value))
  test_output<-data.frame(df,p_value)
  names(test_output)<-c("deg_freedom","LB_p_value")
  return(test_output)
}

AR1.LB<-LB_test(resid.AR1,max.k=11,p=1,q=0)
#To see the table of P-values, type
AR1.LB
plot(AR1.LB$deg_freedom,AR1.LB$LB_p_value,xlab="Degrees of freedom",ylab="P-value",main="Ljung-Box test P-values",ylim=c(0,1))
abline(h=0.05,col="blue",lty=2)



# MA(1) model
model.MA1<-arima(annual_mtemp2,order=c(0,0,1),method="ML")
arima(x = annual_mtemp2, order = c(0, 0, 1), method = "ML")
resid.MA1<-residuals(model.MA1)
par(mfrow=c(3,1))
ts.plot(resid.MA1)
acf(resid.MA1,main="Series resid.MA1")
MA1.LB<-LB_test(resid.MA1,max.k=11,p=0,q=1)
#To see the table of P-values, type
```

```
plot(MA1.LB$deg_freedom,MA1.LB$LB_p_value,xlab="Degrees of freedom",ylab="P-value",main="Ljung-Box test P-values",ylim=c(0,1))
abline(h=0.05,col="blue",lty=2)

# MA(2) model
model.MA2<-arima(annual_mtemp2,order=c(0,0,2),method="ML")
arima(x = annual_mtemp2, order = c(0, 0, 2), method = "ML")
resid.MA2<-residuals(model.MA2)
#par(mfrow=c(3,1))
ts.plot(resid.MA2)
acf(resid.MA2,main="Series resid.MA2")
MA2.LB<-LB_test(resid.MA2,max.k=11,p=0,q=2)
plot(MA2.LB$deg_freedom,MA2.LB$LB_p_value,xlab="Degrees of freedom",ylab="P-value",main="Ljung-Box test P-values",ylim=c(0,1))
abline(h=0.05,col="blue",lty=2)


# ARMA(1,1) model
model.ARMA<-arima(annual_mtemp2,order=c(1,0,1),method="ML")
arima(x = annual_mtemp2, order = c(1, 0, 1), method = "ML")
resid.ARMA<-residuals(model.ARMA)
#par(mfrow=c(3,1))
ts.plot(resid.ARMA)
acf(resid.ARMA,main="Series resid.ARMA")
ARMA.LB<-LB_test(resid.ARMA,max.k=11,p=1,q=1)
plot(ARMA.LB$deg_freedom,ARMA.LB$LB_p_value,xlab="Degrees of freedom",ylab="P-value",main="Ljung-Box test P-values",ylim=c(0,1))
abline(h=0.05,col="blue",lty=2)


#Function to calculate the theoretical ACF and PACF
t.acf<-ARMAacf(ar=c(0),ma=c(-0.896),lag.max=30)
t.pacf<-ARMAacf(ar=c(0),ma=c(-0.896),lag.max=30,pacf=TRUE)
par(mfrow=c(1,2))
```

```
#Produce a plot of the sample ACF for the simulated data
acf(annual_mtemp2)
#Add a line (in red) that shows the theoretical ACF against lags 0 to 100
y_lag<-c(0:30)
lines(y_lag,t.acf,col="red")
#Plot the sample PACF
pacf(annual_mtemp2,ylim=c(-1,1))
#Add the theoretical PACF to this plot as a red line
lines(c(1:30),t.pacf,col="red")
```

II. TASK 2 R CODE

```
library(forecast)
em_house_prices <- read_csv("Desktop/20353100-MATH4022-CW/em_house_prices.csv")
avg_house<-ts(em_house_prices$average_price_gbp,start=2010,frequency=12)
tsdisplay(avg_house,main="Time plot of monthly average house prices in the East
Midlands (2010-2019)")

#Removing stationary
s1<-diff(avg_house)
tsdisplay(s1)

#Removing seasonally
s12<-diff(s1,12)
tsdisplay(s12)

#First difference of the seasonally
avg_house_diff1<-diff(avg_house_diff12)
tsdisplay(avg_house_diff1)

#LB SARIMA test
LB_test_SARIMA<-function(resid,max.k,p,q,P,Q){
  lb_result<-list()
  df<-list()
  p_value<-list()
  for(i in (p+q+P+Q+1):max.k){
    lb_result[[i]]<-Box.test(resid,lag=i,type=c("Ljung-Box"),fitdf=(p+q+P+Q))
    df[[i]]<-lb_result[[i]]$parameter
    p_value[[i]]<-lb_result[[i]]$p.value
  }
  df<-as.vector(unlist(df))
  p_value<-as.vector(unlist(p_value))
  test_output<-data.frame(df,p_value)
  names(test_output)<-c("deg_freedom","LB_p_value")
  return(test_output)
}

# ARIMA(1, 1, 1) × (1,1,0) 12
t1<-arima(avg_house,order=c(1,1,1), seasonal=list(order=c(1,1,0),period=12),
method="ML")
arima(x = avg_house, order=c(1,1,1), seasonal=list(order=c(1,1,0),period=12), method =
"ML")
tsdiag(t1)

# ARIMA(1, 1, 1) × (1,1,1) 12
t2<-arima(avg_house,order=c(1,1,1), seasonal=list(order=c(1,1,1),period=12),
method="ML")
arima(x = avg_house, order=c(1,1,1), seasonal=list(order=c(1,1,1),period=12), method =
"ML")
tsdiag(t2)

# ARIMA(1, 1, 1) × (0, 1, 0) 12
t3<-arima(avg_house,order=c(1,1,1), seasonal=list(order=c(0,1,0),period=12),
method="ML")
arima(x = avg_house, order=c(1,1,1), seasonal=list(order=c(0,1,0),period=12), method =
"ML")
tsdiag(t3)

# ARIMA(1, 1, 1) × (0, 1, 1) 12
t4<-arima(avg_house,order=c(1,1,1), seasonal=list(order=c(0,1,1),period=12),
method="ML")
arima(x = avg_house, order=c(1,1,1), seasonal=list(order=c(0,1,1),period=12), method =
"ML")
tsdiag(t4)

# ARIMA(1, 1, 2) × (0, 1, 1) 12  best model
modelbest<-arima(avg_house,order=c(1,1,2), seasonal=list(order=c(0,1,1),period=12),
method="ML")
```

```
arima(x = avg_house, order=c(1,1,2), seasonal=list(order=c(0,1,1),period=12), method =
"ML")
resid<-residuals(modelbest)
par(mfrow=c(3,1))
ts.plot(resid)
acf(resid,main="Series resid")
modelbest.LB<-LB_test_SARIMA(resid,max.k=12,p=1,q=2,P=0,Q=1)
modelbest.LB
plot(modelbest.LB$deg_freedom,modelbest.LB$LB_p_value,xlab="Degrees of
freedom",ylab="P-value",main="Ljung-Box test P-values",ylim=c(0,1))
abline(h=0.05,col="blue",lty=2)

# ARIMA(1, 1, 2) × (0,1,0) 12
t5<-arima(avg_house,order=c(1,1,2), seasonal=list(order=c(0,1,0),period=12),
method="ML")
arima(x = avg_house, order=c(1,1,2), seasonal=list(order=c(0,1,0),period=12), method =
"ML")
tsdiag(t5)

# ARIMA(1, 1, 2) × (1,1,1) 12
t6<-arima(avg_house,order=c(1,1,2), seasonal=list(order=c(1,1,1),period=12),
method="ML")
arima(x = avg_house, order=c(1,1,2), seasonal=list(order=c(1,1,1),period=12), method =
"ML")
tsdiag(t6)

# ARIMA(1, 1, 2) × (1， 1， 0) 12

t7<-arima(avg_house,order=c(1,1,2), seasonal=list(order=c(1,1,0),period=12),
method="ML")
arima(x = avg_house, order=c(1,1,2), seasonal=list(order=c(1,1,0),period=12), method =
"ML")
tsdiag(t7)
```

```
# ARIMA(2, 1, 1) × (1， 1， 0) 12

t8<-arima(avg_house,order=c(2,1,1), seasonal=list(order=c(1,1,0),period=12),
method="ML")
arima(x = avg_house, order=c(2,1,1), seasonal=list(order=c(1,1,0),period=12), method =
"ML")
tsdiag(t8)

# ARIMA(2, 1, 1) × (0， 1， 1) 12

t9<-arima(avg_house,order=c(2,1,1), seasonal=list(order=c(0,1,1),period=12),
method="ML")
arima(x = avg_house, order=c(2,1,1), seasonal=list(order=c(0,1,1),period=12), method =
"ML")
tsdiag(t9)

# ARIMA(2, 1, 1) × (0， 1， 0) 12

t10<-arima(avg_house,order=c(2,1,1), seasonal=list(order=c(0,1,0),period=12),
method="ML")
arima(x = avg_house, order=c(2,1,1), seasonal=list(order=c(0,1,0),period=12), method =
"ML")
tsdiag(t10)

# ARIMA(2, 1, 1) × (1， 1， 1) 12

t11<-arima(avg_house,order=c(2,1,1), seasonal=list(order=c(1,1,1),period=12),
method="ML")
arima(x = avg_house, order=c(2,1,1), seasonal=list(order=c(1,1,1),period=12), method =
"ML")
tsdiag(t11)

#Forecast for the next 6 months (h=6) from the fitted
fc_6m<-forecast(avg_house,h=6,model=modelbest,level=95)
fc_6m
#Plot the forecasted values
```

```
autoplot(fc_6m)

#Assuming we don't know the data for 2019 to forcast
x<-ts(em_house_prices$average_price_gbp,start=2010,end=2019,frequency=12)
mo<-arima(x,order=c(1,1,2), seasonal=list(order=c(0,1,1),period=12), method="ML")
fc_2019<-forecast(x,h=6,model=mo,level=95)

#We extract the values of the series for 2017 and 2018
price17_18<-ts(avg_house[86:109],start=2017,frequency=12)

#We read in the true number of average_price in 2019 as a time series of length 6
Actual2019<-
ts(c(191237,190990,190586,191571,191927,192029),frequency=12,start=2019)

#Using the model and forecasts fitted earlier combine all series into one object, for plotting
price_fit<-cbind(fc_2019$mean,fc_2019$upper,fc_2019$lower,price17_18,Actual2019)

#Plot the true values for 2017 and 2018 and the forecasted and true values for the first six
months of 2019 (with 95% conf. int.)
autoplot(price_fit)
```