Time Series and Forecasting - Coursework Assignment

David Hui Ka Leuk (20490457)

2023-04-25

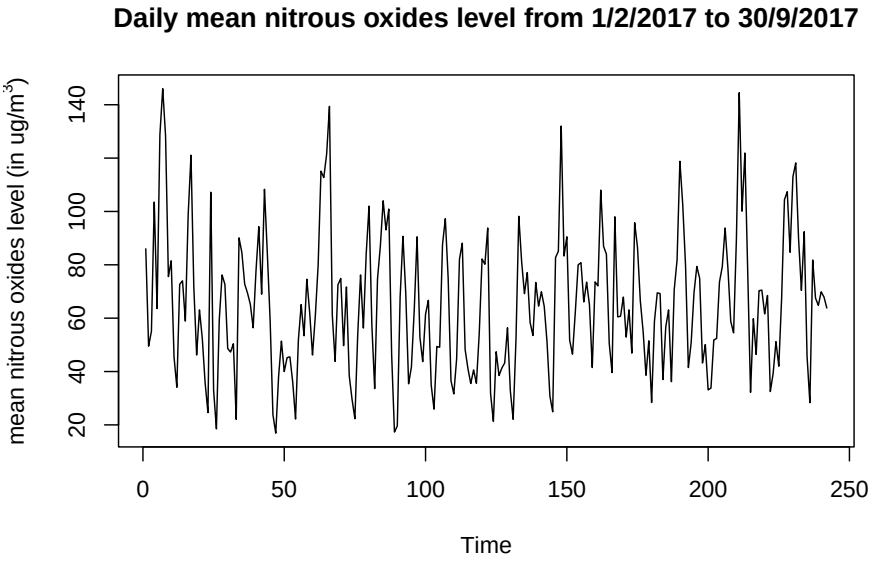
Question 1

This task is to analyse the given dataset by fitting a suitable time series model to describe these data.

Overview

The given dataset contains 242 daily mean measurement of the nitrous oxides level (in ug/m^3), recorded at a location on the A23 Purley Way road in the London Borough of Croydon, from 1/2/2017 to 30/9/2017 (inclusively).

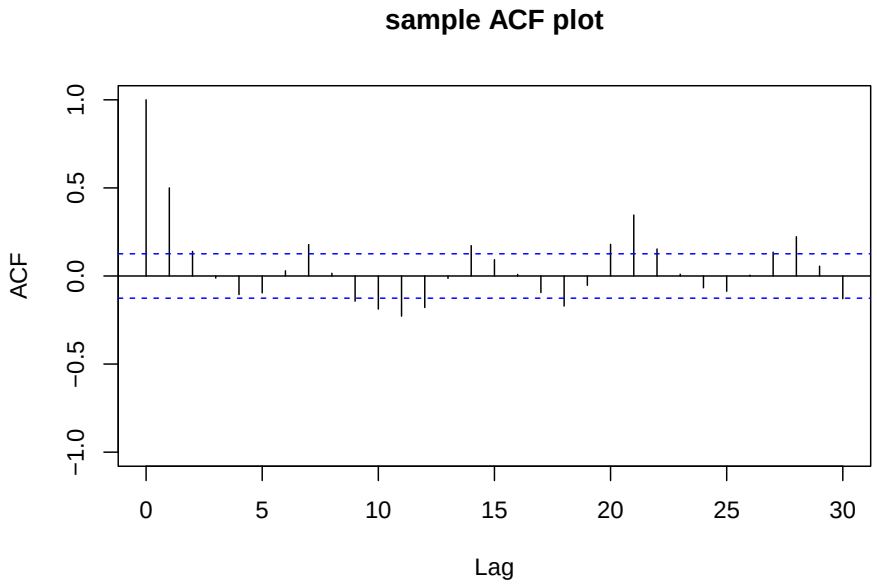
First import the dataset and then look at the time-series plot



From the time-series plot, it appears

1. no trends and seasonal components.
2. have constant mean
3. although some period appears to have smaller variance (x-axis around 50) and some variability looks larger (around 150). However, it just looks random fluctuation only. So we may treat the data to have constant variance (no systematic change in variance over time)

We can then look at the sample ACF plot



From the sample ACF plot, we can spot that, at lags multiple of 7, the sample ACF value lie outside the bound $\pm\frac{2}{\sqrt{n}}$. However, their values are still quite close to the bound (except for lag 21). So I would continue to think that the sample ACF are close to zero as lags increase, and not consider that these situation come from any seasonal behavior.

Now, since the sample ACF looks decay to zero as lags increase.

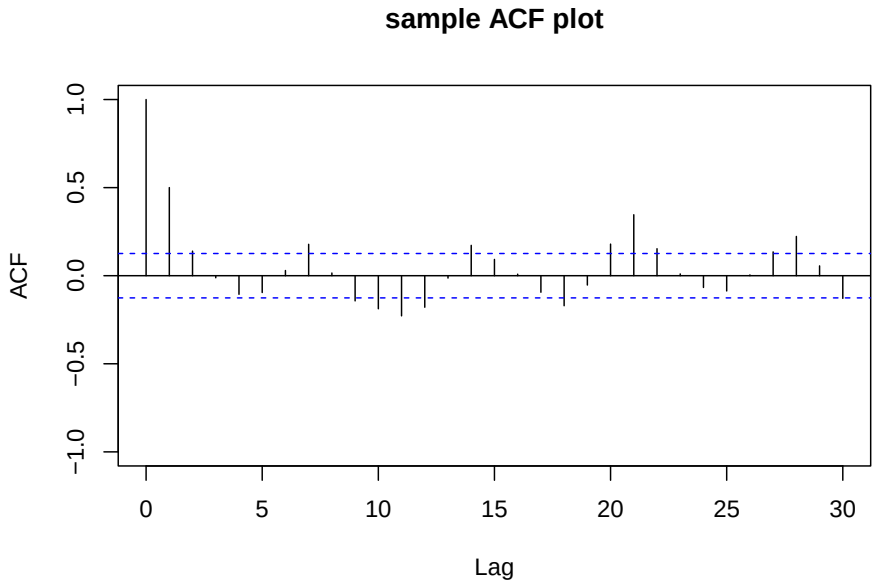
Because I think that there are no systematic change in mean and variance, and the ACF close to zero as lags increase, it seems plausible to claim that the given time series data is **stationary**

So, we can try to fit an ARMA model to the data.

Identify the order of the process

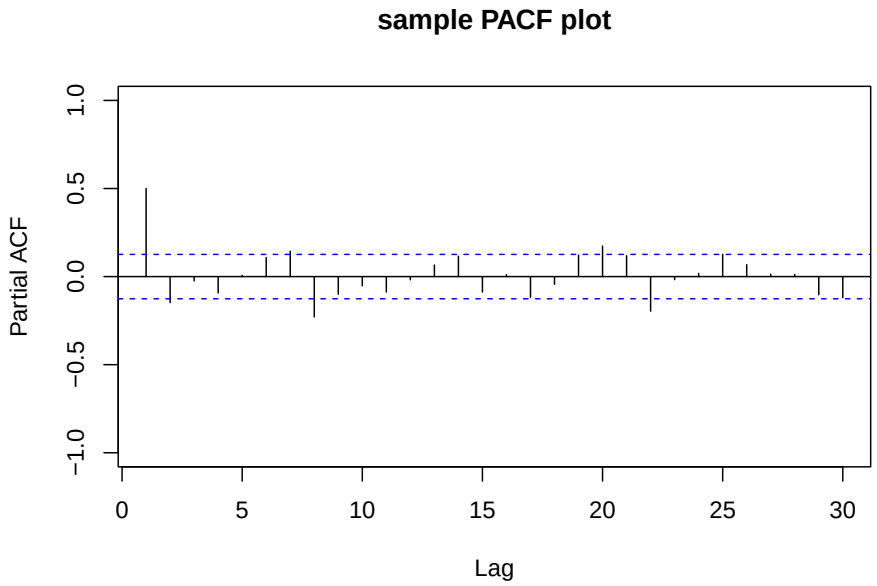
Initial inspection

Look at the ACF plot again



there does not have obvious cut-off after some lag. So this may suggest that the process includes AR

components.



From the sample PACF plot, there seems a obvious cut-off after lag 1. It suggests that the data may come from a AR(1) process.

So I will start fitting the data using AR(1) first. Then try to include more parameters to see if they may have significant improvement.

Let X_t be the original process with mean μ

Fitting AR(1)

$$X_t - \mu = \phi_1(X_{t-1} - \mu) + Z_t \ ; \ var(Z_t) = \sigma_z^2$$

```
# fit the data using arima function, with AR order=1
arima(data, order = c(1, 0, 0), method = "ML")
```

```
##
## Call:
## arima(x = data, order = c(1, 0, 0), method = "ML")
##
## Coefficients:
##          ar1  intercept
##       0.4994    65.3338
## s.e.  0.0555     2.8826
##
## sigma^2 estimated as 508.1:  log likelihood = -1097.44,  aic = 2200.87
```

Consider the test of hypothesis:

$$H0 : \phi_1 = 0$$
$$H1 : \phi_1 \neq 0$$

since our estimate $= 0.4994$ with s.e. $= 0.0555$, the test statistic $= |\frac{0.4994}{0.0555}| = 8.9981982 > 2$
So we can conclude that ϕ_1 is significant, and should be kept into the model

Try fitting higher orders of ARMA(p,q)

To try fitting different orders, each time I add one more parameter to the model, and test if the newly parameter is significant or not. I then keep trying until I find the newly added parameter is not significant. Note that this procedure may produce the problem of multiple testing. However, this should be helpful for me to find "possible" model first.
After I found a list of model candidates, I will then use AIC to determine the best model among them. Since smaller AIC would indicate a better model, so I will compare the AIC of different fitted models, and see whether or not introducing more parameters will have significant improvement.

After running the same code for different AR and MA orders, I end up with the following orders:

model	AIC
AR(1)	2200.872
AR(2)	2197.361
MA(1)	2201.435
MA(2)	2198.251
ARMA(1,1)	2197.787

From this summary, firstly, we can see that the AIC of MA(1) is larger than that of AR(1), so we can conclude that MA(1) may not better than AR(1). To compare MA(2), AR(2) and ARMA(1,1), those give lower AIC than AR(1), so they may indicate that these 3 models have better fit.
Since AR(2) gives the lowest AIC among them, and the drop seems reasonable (usually, if adding one more parameter to just give a drop 0.1 in AIC, then by the principle of parsimony, choosing a model with fewer parameter is better. But now the drop in AIC is 3, I think using AR(2) should be reasonable)

Chosen Model: AR(2)

My chosen model for this dataset is an AR(2), and we can extract the parameters from the fitted model, round up to 4 significance figures

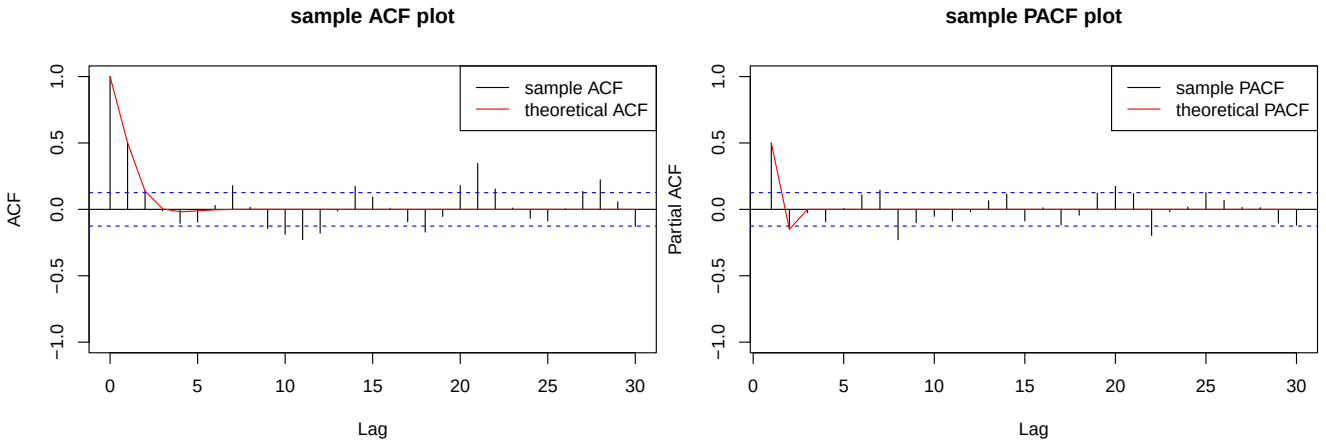
My suggested model is

$$(X_t - 65.33) = 0.5753\ (X_{t-1} - 65.33) - 0.1501\ (X_{t-2} - 65.33) + Z_t\ ;\ var(Z_t) = 496.6$$

Model Diagnostic

ACF and PACF plot with overlay theoretical ACF and PACF

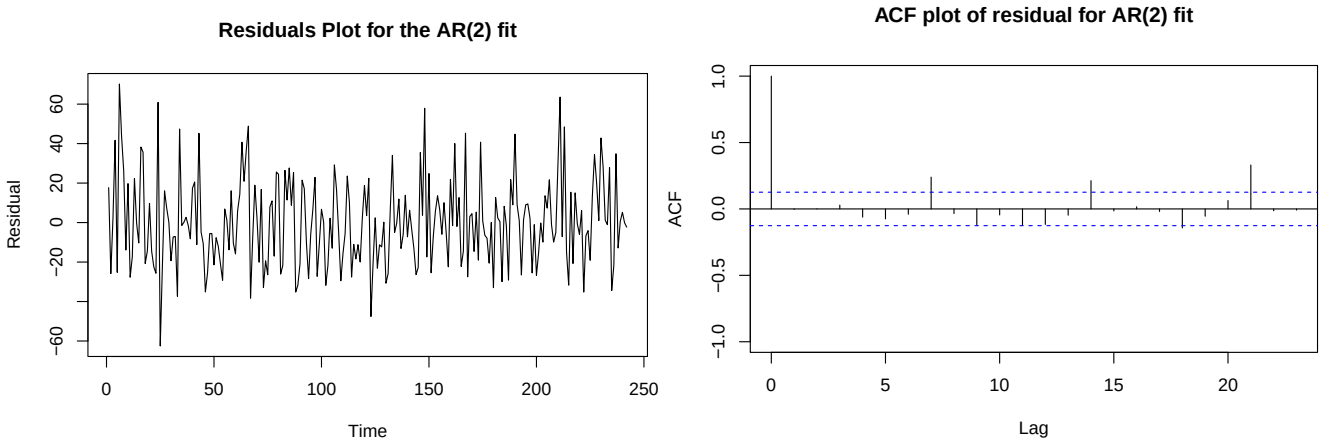
The sample ACF and PACF plot with overlay the ACF and PACF using our AR(2) model



We can see that both ACF and PACF are fitting quite good using the chosen model

Residuals plot and Ljung-Box test

We can extract the residuals from model fit, and then plot the time-series plot and ACF plot

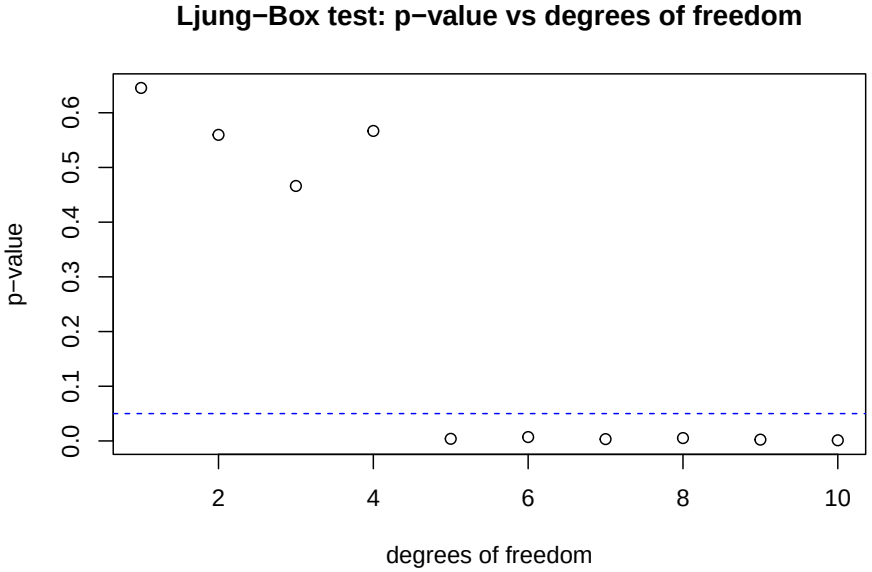


From the residuals plot and the ACF plot, they look fine like a White-noise process. Although for the ACF plot, there again appears to have spikes at lags 7, 14, 21 (multiple of 7). However as the ACF values are quite small and close to the bound. I would still treat them as not significant.

Now, lets perform Ljung-Box test, for degrees of freedom from 1 to 10

$$H0 : \text{residuals are independent}$$
$$H1 : \text{residuals are dependent}$$

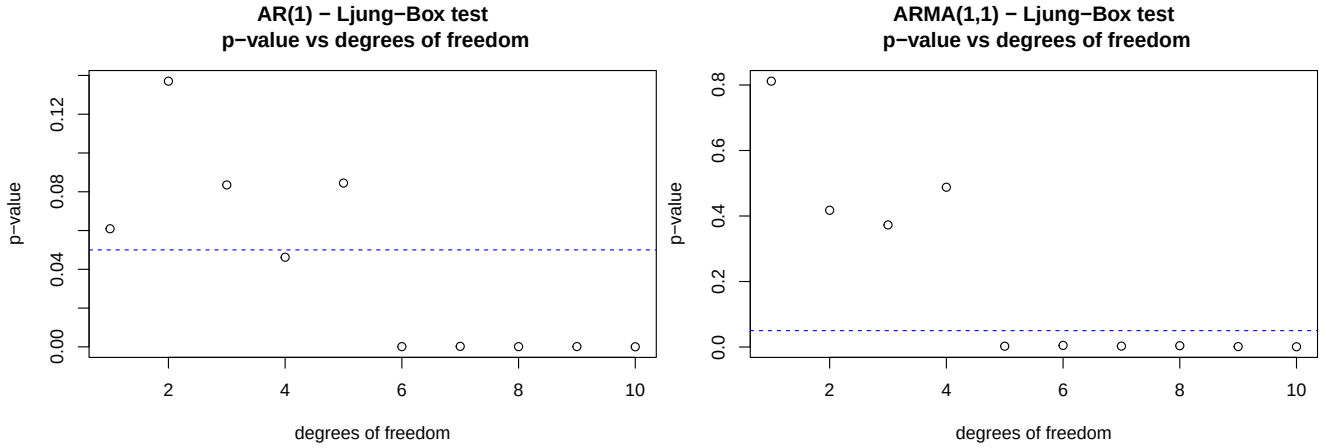
And if the residuals are independent, we can see that from the Ljung-Box test, for each degrees of freedom, the p-value would be larger than 0.05 in order to retain H0.



Here, we can see that, for degrees of freedom (df) ≥ 5, the p-value of the test is less than 0.05. So this gives that the residuals are correlated for large df. Since we failed to conclude that the model AR(2) satisfied some model assumption: $\{Z_t\}$ are white noise process, so we need to go through further discussion.

Discussion

I have tried to perform the same diagnostic procedures for those model candidates (i.e. MA(1), MA(2), AR(1), AR(2), and ARMA(1,1)), and I found that all of them also conclude that the residuals are correlated. For example, see the Ljung-Box test for AR(1) and ARMA(1,1):



Hence, these models failed to satisfied uncorrelated residuals as well.

Although we can try more complicated model like ARIMA and SARIMA (because there seemed to have slight correlation with period = 7). However, since we cannot find obvious evidence (pattern) to believe that we should use those. Because AR(2) overall behavior is good, but just failed in Ljung-Box test. By the principle of parsimony, I would rather accept AR(2) to be my selected model, than increase the complexity of the model.

Conclusion

According to these information, I think AR(2) model is a plausible model to fit our data even though it failed to comply with uncorrelated residuals assumption. My chosen model is AR(2):

$$(X_t - 65.33) = 0.5753\ (X_{t-1} - 65.33) - 0.1501\ (X_{t-2} - 65.33) + Z_t\ ;\ var(Z_t) = 496.6$$

Question 2

This task is to analyse the number of new cars registered in England for each quarter (of a year) and forecast numbers of new cars registered in subsequent 4 quarters.

Executive Summary

Overview

The given dataset contains data from 2001 Q1 to 2022 Q3. There are 87 records, with average number 496.808. We can find that the number of new cars registration (in thousands) shows a seasonal pattern, where generally high in Q1, then a slight drop in Q2, followed by a rise in Q3, and finally a larger drop and reach to the lowest of the year in Q4.

Outliers

There are 2 obvious outliers with particularly low values. One is in 2008 Q3, another one is in 2020 Q2. We cannot certainly know the exact reason. Possible reason is the financial tsunami in 2007-08; and COVID-19 pandemic in 2019-20. Both events may largely impact on the number of new cars registration.

Model

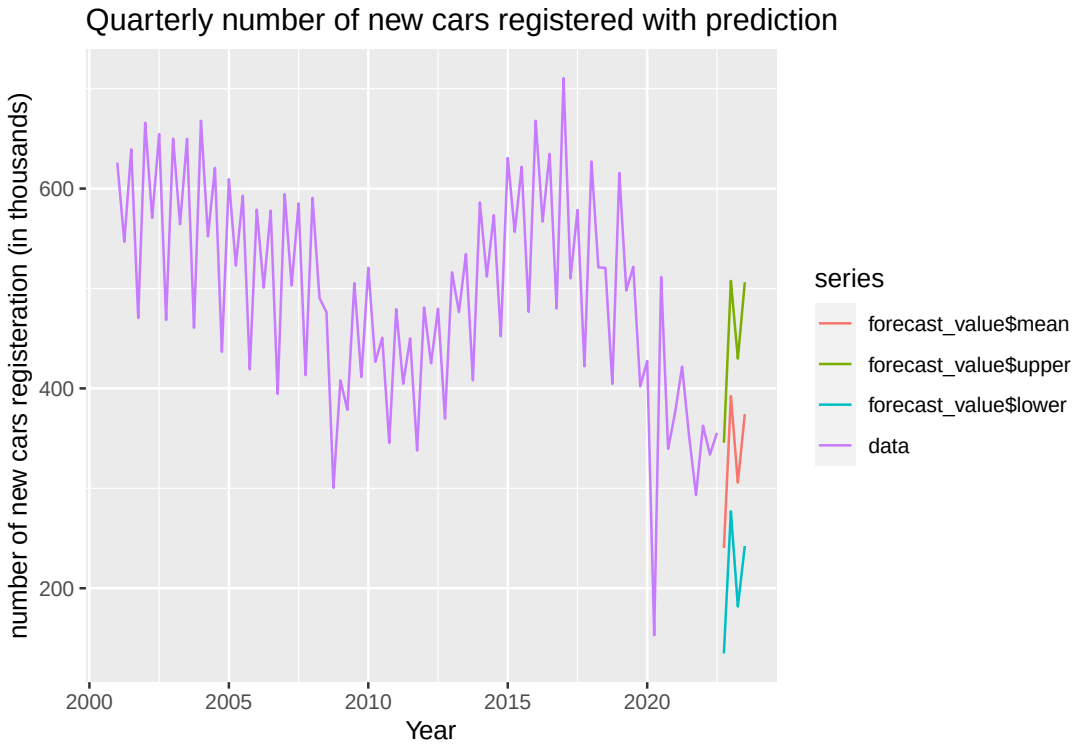
From the analysis, I would fit a ARIMA(1,0,1)(0,1,1)[4]. let $\{X_t\}$ denote the process of the number of new cars registration; let $\{Z_t\}$ denote the white noise process with variance σ_Z^2, then the fitted model:

$$(1-\hat{\phi}_1 B)(1-B^4)(X_t-\hat{\mu}) = (1+\hat{\theta}_1 B)(1+\hat{\Theta}_1 B^4)Z_t$$

where $\hat{\mu} = 496.808,\ \hat{\phi}_1 = 0.98268,\ \hat{\theta}_1 = -0.5387,\ \hat{\Theta}_1 = -0.9318,\ \hat{\sigma_Z^2} = 2878$

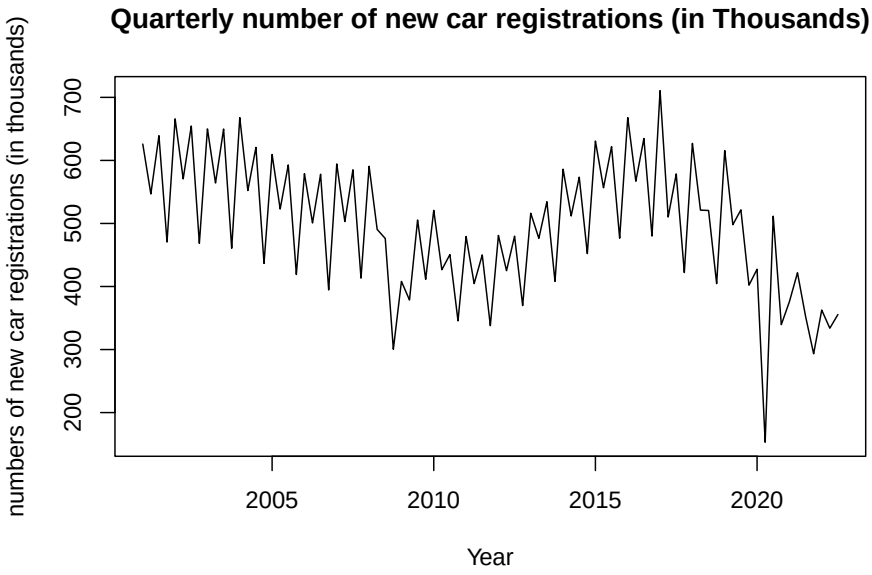
Forecasting

The forecasted numbers of new cars registered (in thousand) in the next 4 quarters with uncertainties (95% confidence level)



Descriptive Information

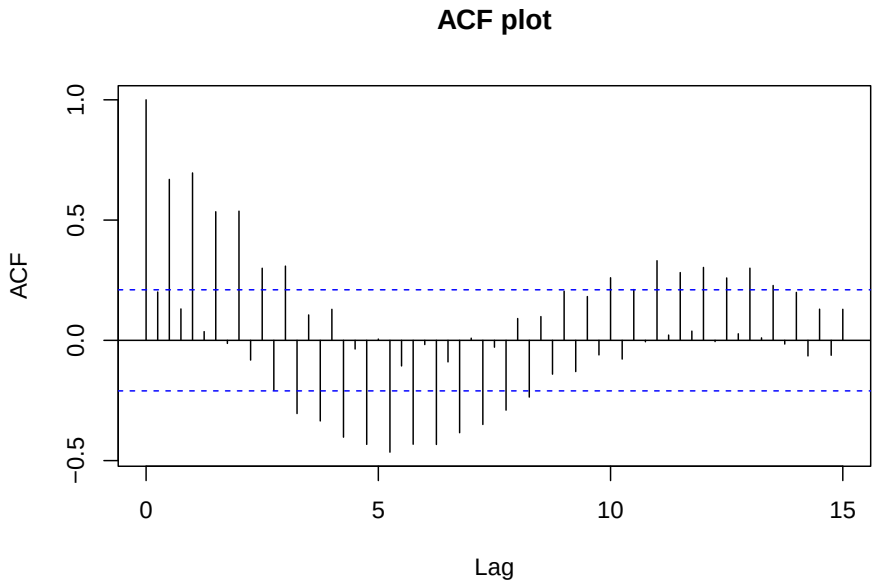
The given dataset contains data from 2001 Q1 to 2022 Q3. There are 87 records, with average number 496.808.



From the time-series plot above, we can identify the followings:

1. seasonality: a "up-and-down" seasonal pattern of period 4. Each year, the value would be higher in Q1; then a slight drop in Q2; after that a rise in Q3; and finally a larger drop in Q4. Then a large rise in the next Q1, and repeat this pattern.
2. trend: the variability apparently the same for different period, but the mean are clearly changed over different period: from 2001 to 2010, it appears to have a decreasing trend linearly; from 2010 to 2017, it appears to have a increasing trend linearly; finally from 2017 and afterwards, it showed a decreasing trend again
3. outliers: it appears to have obvious outliers (in 2008 Q3, and 2020 Q2). In these 2 quarters, it has a large drop. Although we cannot certainly find out the reason. Possible events are financial tsunami in 2007-08, and COVID-19 pandemic in 2019-20. These events may have a large impact on new car registrations.

Furthermore, we can also look at the ACF plot

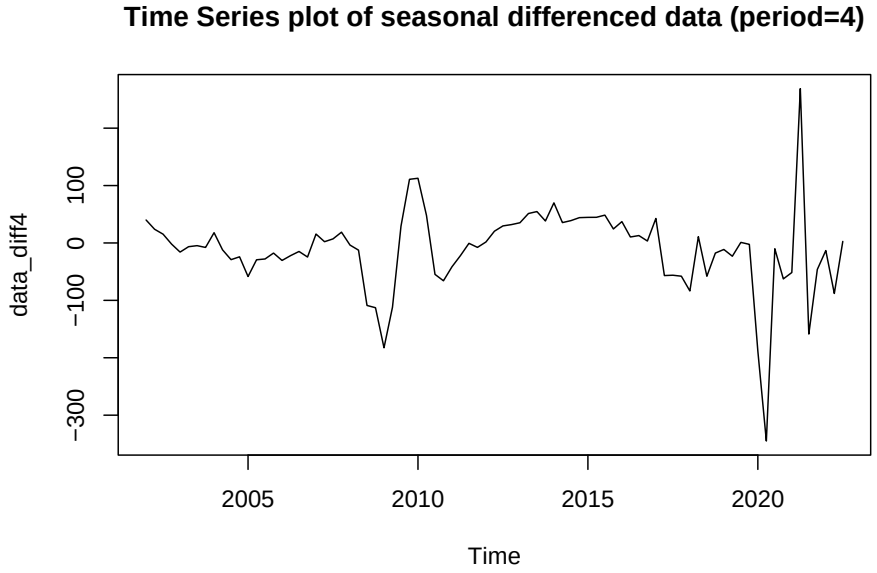


Since the sample ACF does not getting close to 0 as lags increase, which is also a sign of non-stationarity.

Because of these reasons, we can conclude that the given data is **non-stationary**

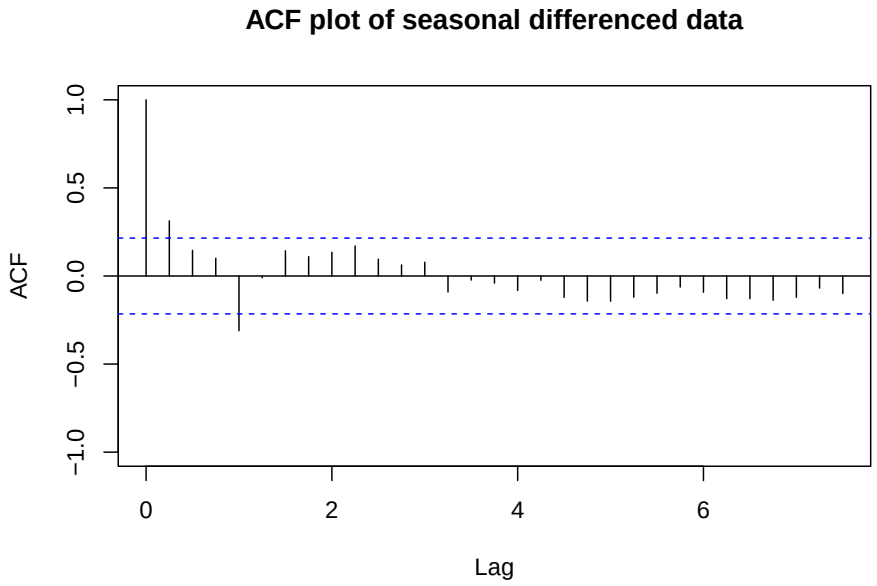
Assessing Stationarity

Because there are a seasonal pattern of period 4, we can perform seasonal differencing and give the plot



Although there are some rapid fluctuation around time=2010 and time=2020. However, we might just view that as random fluctuation because the rest part of the plot looks stationary because there mean and variance does not have systematic changes over time. Because there are no obvious trend in this time plot, so I do not need to perform 1st order differencing. Also, because I think that the variability does not change over time, so I do not need to perform any transformation as well.

Next, We can see the ACF of the seasonal differenced data to see if the seasonal differenced data have stationary behavior

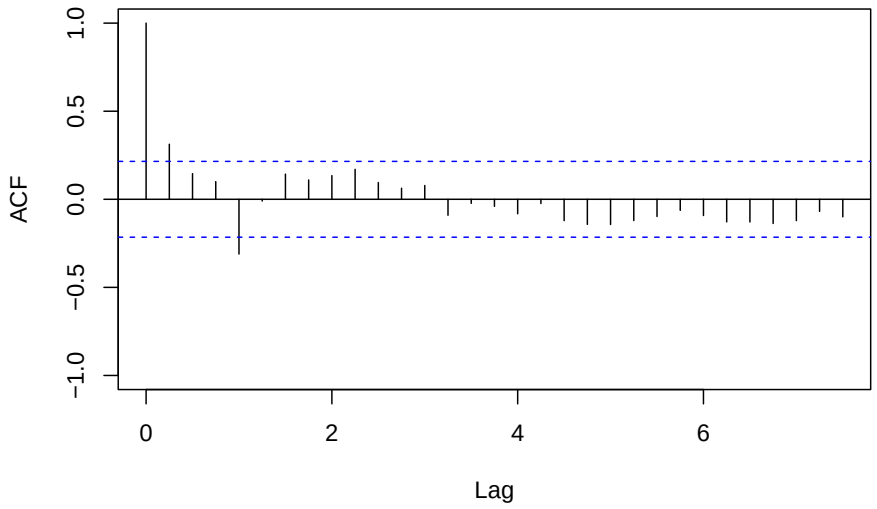


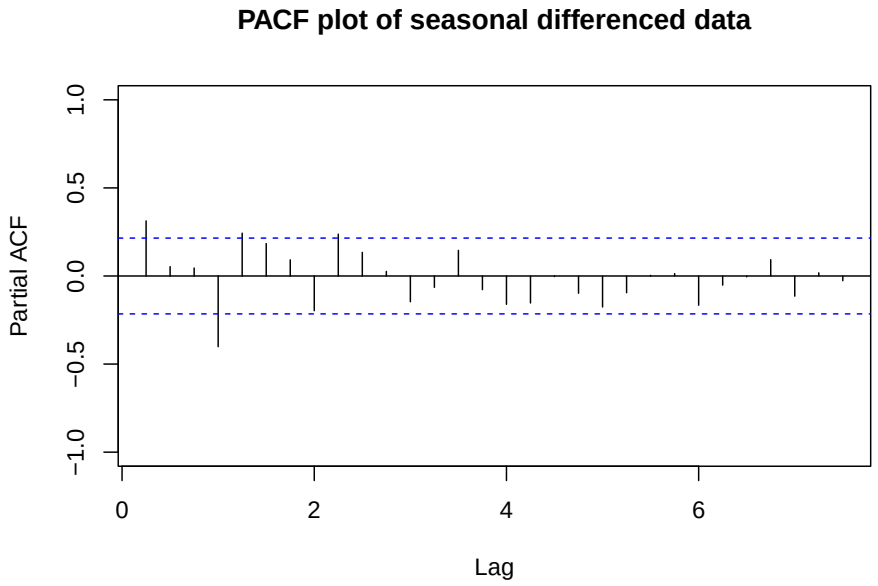
From the sample ACF plot above, we can see that the ACF value approaches to 0 as lag increases, which also give a usual behavior of a stationary time series (Note that for seasonal differenced ACF plot, for the x-axis, Lag=1 means a seasonal period, so you will see that there are 8 values between Lag=0 (exclusive) to Lag=2 (inclusive))

Hence according to these information, I may conclude that seasonal differencing give a stationary time-series.

Model Indenification

To identify the order of an ARMA(p,q) process, the usual way is to look at the sample ACF plot and sample PACF plot to see if there are any obvious cut-off after specific lag.





By looking at both ACF and PACF plot, there do not seem to have a clear cut-off after specific lag. So it may indicate that the underlying process contain both AR and MA component.

Model fitting and selection

In general, we would fit the original data using **arima** method directly by supplying non-seasonal order and seasonal order, rather than fit an ARMA model to the seasonal differenced data.

Also by the general rules of thumb, we would prefer SARIMA with each orders not greater than 2. Hence, I will start with purely simple model ARIMA(0,0,0)(0,1,0)[4] first, then procedurally increase the order of each seasonal and non-seasonal components until 2, to fit the data in order to get the AIC.

There will be run through 3^4=81 number of iterations. Note that when some of the parameter estimate fall outside admissible range (we need parameter estimate within $\pm$ 1 to ensure stationarity and invertibility). In this case, my method would simply ignore those.

From this greedy search, I find that there are 76 (out of 81) fit properly. The following table shows different iteration with the smallest 10 AIC

p	q	P	Q	AIC
1	1	0	1	911.0427
1	2	0	1	913.0342
2	1	0	1	913.0359
1	1	1	1	913.0379
1	1	0	2	913.0384
1	1	1	2	914.8513
1	2	1	1	915.0325
1	2	0	2	915.0328
2	1	0	2	915.0339
2	1	1	1	915.0339

Where p is non-seasonal AR order; q is non-seasonal MA order; P is seasonal AR order; Q is seasonal MA order. Since from model selection criteria, generally lower AIC means a better model. From the table

above, we can see that the lowest AIC = 911.0427, the corresponding model is ARIMA(1,0,1)(0,1,1)[4] (for short, name this model as M_1)

It is worth to point out that, among these 10 records, M_1 is not only with the smallest AIC, but also with the least numbers of order (nonseasonal orders + seasonal orders = 3), whereas the rest 9 are at least 4.

This gives a good indicator that this M_1 should be capable to represent our data well.

Our model is:

$$(1-\phi_1 B)\nabla_4(X_t-\mu) = (1+\theta_1 B)(1+\Theta_1 B^4)Z_t$$
$$with\ var(Z_t) = \sigma_Z^2$$

Note that ∇_4 is equivalent to $1-B^4$

To get the estimates of the parameters, look at the fitted summary

```
##
## Call:
## arima(x = data, order = c(1, 0, 1), seasonal = list(order = c(0, 1, 1), period = 4),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1     sma1
##       0.9845  -0.5401  -0.9360
## s.e.  0.0548   0.0969   0.1561
##
## sigma^2 estimated as 2878:  log likelihood = -451.52,  aic = 911.04
```

we can get

$$\hat{\phi}_1 = 0.9845$$
$$\hat{\theta}_1 = -0.5401$$
$$\hat{\Theta}_1 = -0.9360$$
$$\hat{\sigma_Z^2} = 2878$$

This fitted summary did not give us $\hat{\mu}$, but we can use the sample mean of the raw data by

$$\hat{\mu} = 496.808$$

In addition, this summary also provides other information for us to look at the significance of the parameters, the test statistic = estimated value / standard error:

```
##       ar1       ma1      sma1
## 17.976106 -5.571953 -5.995092
```

Apart from using AIC alone, for instance, if we wanted to compare M_1 with model having less number of parameter like ARIMA(0,0,1)(0,1,1)[4], we can construct a hypothesis test:

$$H0 : \phi_1 = 0\ vs\ H1 : \phi_1 \neq 0$$

since our test statistic = $|17.9761| > 2$, conclude that ϕ_1 should be kept into the model. And hence, we can claim that we should choose M_1 over ARIMA(0,0,1)(0,1,1)[4]

Model Diagnostic

For model diagnostic, we need to check a few things:

1. theoretical ACF and PACF close to sample ACF and PACF;
2. residuals plot look like a white noise process;
3. residuals ACF plot should be cut-off after lag 0;
4. using Ljung-Box test to test if the residuals are independent.

ACF and PACF plot of the seasonal differenced data

It is not easy to find the sample ACF and PACF of a SARIMA model, but there are R build-in function to calculate the acf and pacf of an ARMA model, using **ARMAacf()** function.

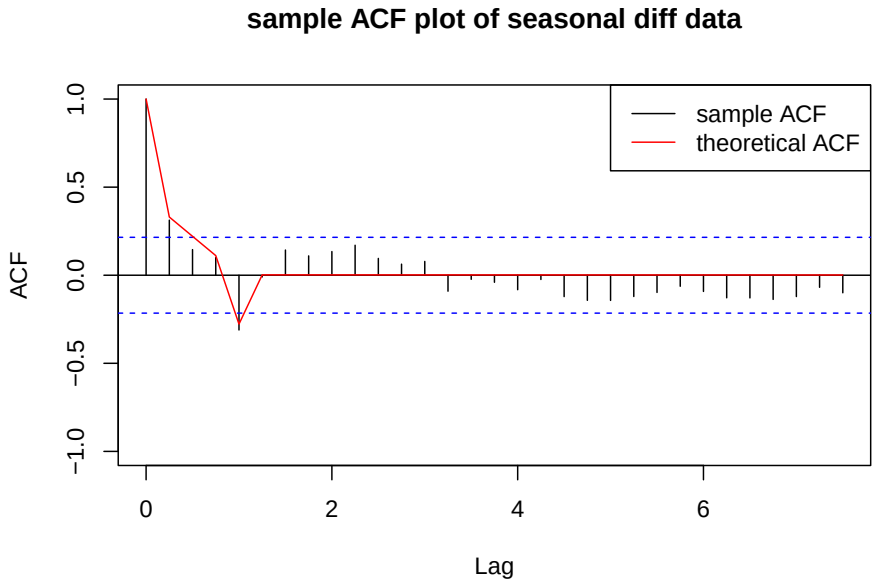
Note that we can view our SARIMA model as an ARMA model of an seasonal differenced data

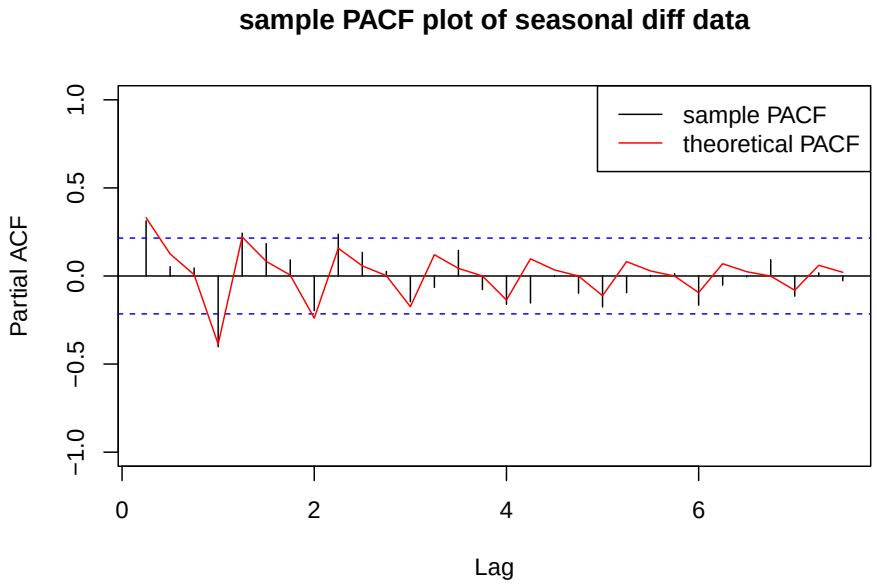
$$\begin{aligned}(1-\phi_1 B)\nabla_4(X_t-\mu) &= (1+\theta_1 B)(1+\Theta_1 B^4)Z_t \\ &= (1+\theta_1 B+\Theta_1 B^4+\theta_1\Theta_1 B^5)Z_t\end{aligned}$$

Hence, our seasonal differenced model $\nabla_4(X_t-\mu)$, can be viewed as an ARMA(1, 5) model. So we can calculate the ACF and PACF, by supplying

$$\begin{aligned}ar &= c(\phi_1), \\ ma &= c(\theta_1, 0, 0, \Theta_1, \theta_1\Theta_1)\end{aligned}$$

Then we can plot the ACF and PACF of the seasonal differenced data, with overlay theoretical ACF and PACF

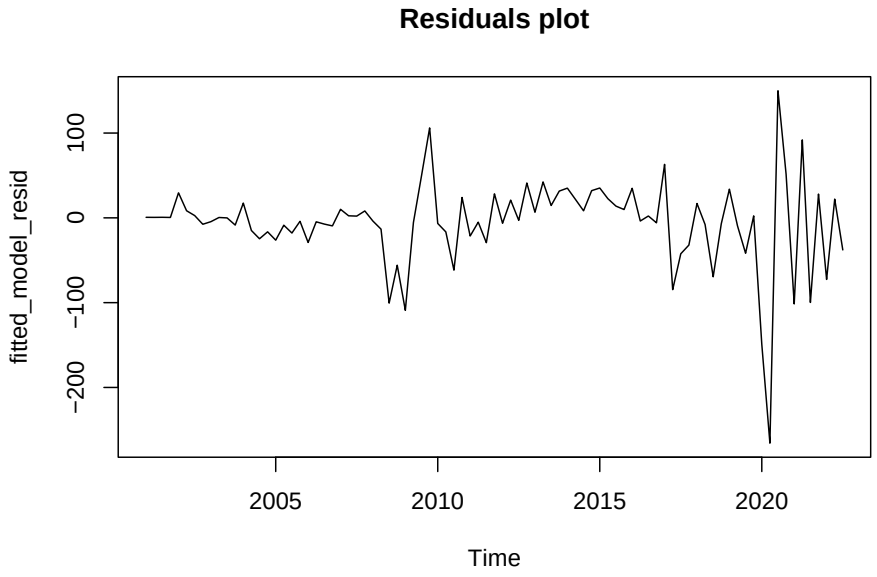


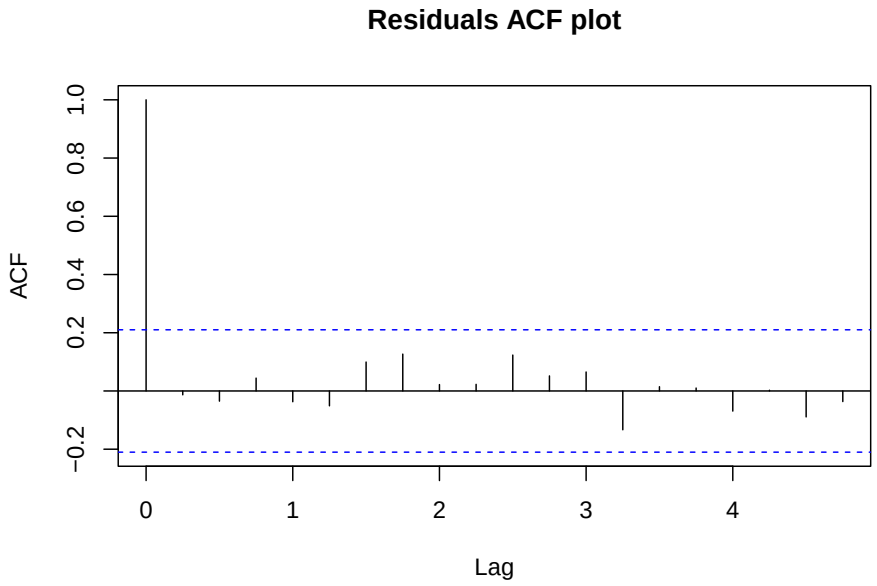


These plots show that our model behave quite well.

Residuals time plot and ACF plot

We can extract the residuals from the model fit through **residuals()** function, and then plot its time plot and ACF plot. For a well-behaved fitted model, the residuals plot should look like a white noise, and the ACF plot should have clear cut-off after lag 0. So I think it is reasonable to think that the residuals look like a white-noise process





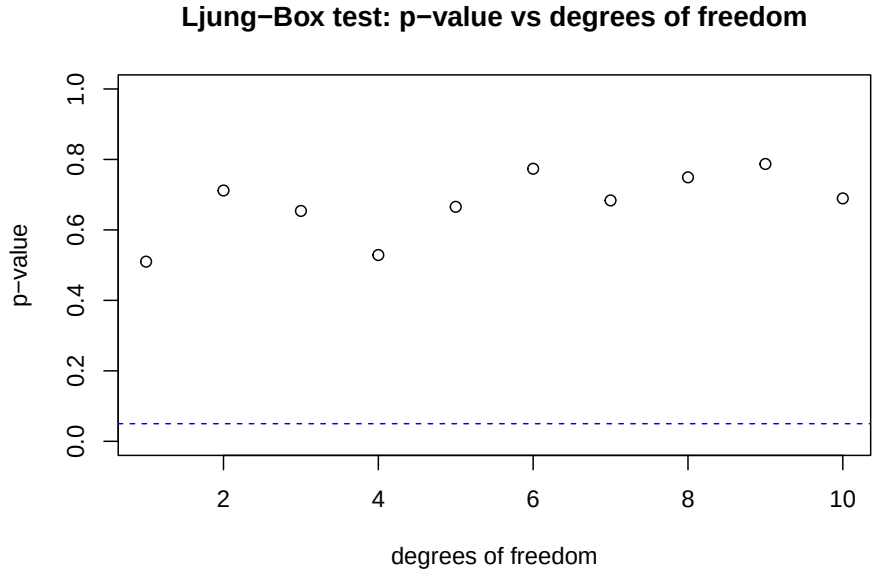
Although the time plot looks a bit fluctuating around time=2010 and time=2020, the ACF plot clearly shows the property of a white-noise process: clear cut-off after lag = 0.

Ljung-Box test

The Ljung-Box hypothesis testing

$$H0 : \text{residuals are independent}$$
$$H1 : \text{residuals are dependent}$$

And if the residuals are independent, we can see that from the Ljung-Box test, for each degrees of freedom (I will perform from 1 to 10), the p-value would be larger than 0.05 in order to retain H0.



Since all p-values of the test lie above 0.05 (the dashed blue line), so we can conclude that for each degrees of freedom, we retain H0. And hence, we can conclude that the residuals are independent.

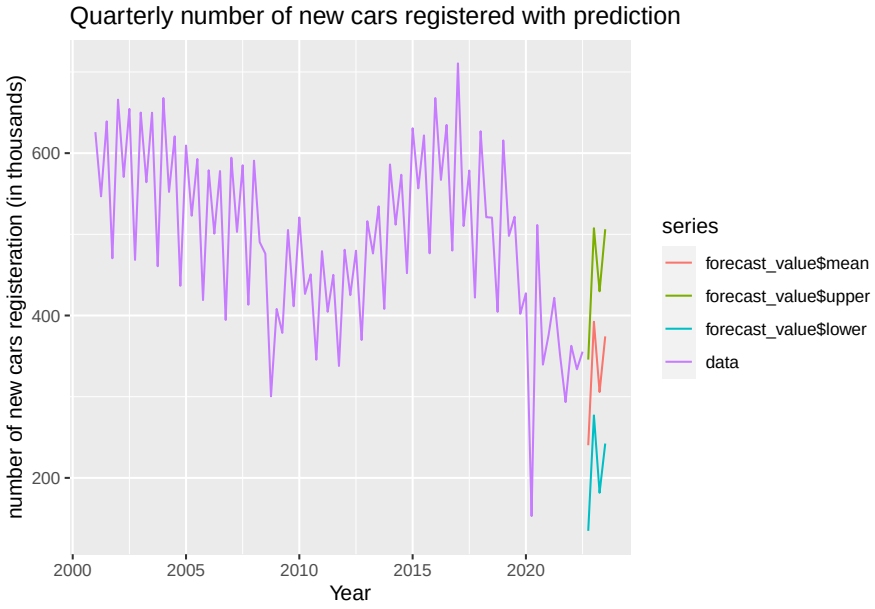
So, from our model diagnostic, there are sufficient evidence to believe that our model is a satisfactory model.

Forecasting

Using **forecast()** from **forecast** library, we can easily forecast the future value at different future time step, together with associated uncertainties. Here I also calculate that 95% confidence interval for each prediction.

$$
\begin{aligned}
&2022\ Q4 : 240.2288 \pm 105.4651 = (134.7638, 345.6939)\\
&2023\ Q1 : 392.2740 \pm 115.2980 = (276.9760, 507.5720)\\
&2023\ Q2 : 305.7431 \pm 124.0725 = (181.6706, 429.8156)\\
&2023\ Q3 : 374.2759 \pm 131.9920 = (242.2839, 506.2679)
\end{aligned}
$$

We can see the original time plot, together with the future forecasting value with associated uncertainties.



Conclusion

After analysing this dataset, we can make the following conclusions:

1. The given dataset contains 87 records, shows seasonal pattern and outliers
2. We can model the given data using a seasonal ARIMA model

$$(1 - \hat{\phi}_1 B)(1 - B^4)(X_t - \hat{\mu}) = (1 + \hat{\theta}_1 B)(1 + \hat{\Theta}_1 B^4) Z_t$$

   $where\ \hat{\mu} = 496.808,\ \hat{\phi}_1 = 0.98268,\ \hat{\theta}_1 = -0.5387,\ \hat{\Theta}_1 = -0.9318,\ \hat{\sigma_Z^2} = 2878$

3. The predicted values with uncertainties are

$$
\begin{aligned}
&2022\ Q4 : 240.2288 \pm 105.4651 = (134.7638, 345.6939)\\
&2023\ Q1 : 392.2740 \pm 115.2980 = (276.9760, 507.5720)\\
&2023\ Q2 : 305.7431 \pm 124.0725 = (181.6706, 429.8156)\\
&2023\ Q3 : 374.2759 \pm 131.9920 = (242.2839, 506.2679)
\end{aligned}
$$

APPENDIX for Q1

```
############## Setup, Graph plotting and basic identification #############

# remove existing environment variables
rm(list=ls())

# set working directory
setwd("/Users/davidhui/Documents/GitHub/uon/MATH4022 Time Series and Forecasting/")

# import the a23_nox.csv
raw_data = read.csv("materials/a23_nox.csv")
head(raw_data)  # see the first few rows
n = nrow(raw_data)  # number of observations
n

# convert into ts object
data = ts(raw_data$daily_mean_nox)

mean(data)  # mean of the data is 65.26074

# plot the row ts plot
ts.plot(data, main="Daily mean nitrous oxides level from 1/2/2017 to 30/9/2017",
        xlab="Time",
        ylab=expression(paste("mean nitrous oxides level (in ug/",m^3,")")))

# from the ts plot, there does not appear to have trends and seasonal components;
# also, the plotted data appears to have constant mean and constant variance

par(mfrow=c(2,1))
acf(data, main="sample ACF plot", ylim=c(-1,1), lag.max=30)
pacf(data, main="sample PACF plot", ylim=c(-1,1), lag.max=30)
par(mfrow=c(1,1))

# sample ACF and sample PACF decay to zero gradually as lags increase;
# it seems plausible to claim that the time series is stationary.

# to identify the order of the ARMA(p,q) process

# from the sample ACF plot, there do not have obvious cut-off after some lag
# > so the data appears to include AR components
# from the sample PACF plot, there "seems" a drop in PACF value after lag 1,
# > however, the sample PACF at lag 2 is still slightly lie outside +- 2 /sqrt(n) bound,
# > so this is in doubt to claim the PACF show obvious cut-off.

# Initially, I will suggest and AR(1) first
# and then try to introduce different ARMA model
```

```
# so let {X_t} be the original process
# note that the mean of the process is non-zero,
# so in our arima(...) model fitting, the estimate of the intercept will be non-zero

########### Identify the order of an ARMA(p,q) ###########

#### first take a look at my initial suggested model AR(1)
# (X_t - mu) = phi1 * (X_t-1 - mu) + Z_t ; var(Z_t) = sigma_z^2
arima(data, order=c(1,0,0), method="ML")
# by the hypothesis: H0: phi1 = 0 vs H1: phi1 != 0
# since the estimate = 0.4994 with s.e. = 0.0555
#   => Test Statistic = abs(0.4994/0.0555) = 8.9981 > 2
# we can see that the test statistic for phi1 (ar1) are greater than 2
# and we should keep phi1 into the model

## since replicate these steps are quite time consuming;
## so I am going to write a function to print the fit summary
##    for different AR and MA order
## data = the given time series object
## p = order of AR components
## q = order of MA components
ARMA_fit = function(data, p, q){
  fit = arima(data, order=c(p,0,q), method="ML")
  print("===== fit summary =====")
  print(fit)
  print("===== test statistic =====")
  # coef(fit) returns the parameter estimation;
  # sqrt(diag(vcov(fit))) returns the s.e. of each parameter est.
  test_statistic = (coef(fit) / sqrt(diag(vcov(fit))))
  # no need to print "intercept
  print(test_statistic[1:(p+q)])

  # return the arima fit object, later I will need to extract the AIC
  return (fit)
}

# to see if this works, repeat for AR(1)
# this print the same results
fit_AR1 = ARMA_fit(data, 1, 0)

## try fitting different orders, each time introduce one more parameter
## because of the principle of parsimony
## we shouldn't add too much parameters to the model
## since simple hypothesis is to test the significance of the newly introduced parameter,
##  > given the others are presented
## so I keep trying until I found the newly added parameter is non-significant
## NOTE that this may comes from the problem of multiple testing;
## nevertheless, this should be helpful for me to find "possible" model first
## later I will use AIC to check if
## introducing more parameters will have significant improvement
```

```
# AR(2)
fit_AR2 = ARMA_fit(data, 2, 0)
# phi2 is significant, given phi1 is presented

## AR(3)
fit_AR3 = ARMA_fit(data, 3, 0)
# phi3 is not significant given phi1 and phi2 are presented

# so for pure AR process, I will end up with AR(2)


# next start to fit ARMA
fit_ARMA11 = ARMA_fit(data, 1, 1)
# theta1 is significant, given phi1 is presented

fit_ARMA21 = ARMA_fit(data, 2, 1)
# phi2 is not significant given phi1 and theta1 are presented

fit_ARMA12 = ARMA_fit(data, 1, 2)
# theta2 is not significant given phi1 and theta1 are presented

fit_ARMA22 = ARMA_fit(data, 2, 2)
# throws error
# should because of the parameter estimate fail
# to fall into "appropriate region" for stationarity/invertibility
# so for ARMA process, I will end up with ARMA(1,1)

# try to fit just MA as well
fit_MA1 = ARMA_fit(data, 0, 1)
# theta1 is significant

fit_MA2 = ARMA_fit(data, 0, 2)
# theta2 is significant, given theta1 is presented

fit_MA3 = ARMA_fit(data, 0, 3)
# theta3 is not significant given theta1 and theta2 are presented

# so for pure MA process, I will end up with MA(2)

## combining the above information, we can just compare the AIC
## of AR(1); AR(2); MA(1); MA(2); ARMA(1,1)
## since the sample size is 242; which is not very small;
## so I will just use AIC instead of corrected AIC
aic_df = data.frame(
  model=c("AR(1)", "AR(2)", "MA(1)", "MA(2)", "ARMA(1,1)"),
  AIC = c(AIC(fit_AR1), AIC(fit_AR2), AIC(fit_MA1), AIC(fit_MA2), AIC(fit_ARMA11))
)

aic_df
```

```
# since for model selection procedure; the small AIC indicates a "better" model
# I start with AR(1), where AIC = 2200.872
# MA(1) gives a larger AIC value, so I would not take MA(1) into account,
# and hence MA(2) is not reasonable to consider
# to compare AR(1) with AR(2) and ARMA(1,1), AR(2) gives the smallest AIC.
# although the difference is not very large (2200.872 for AR(1) vs 2197.361 for AR(2))
# however, since phi2 is significant at 5% significant level
# so I think AR(2) is a reasonable model for this dataset

# extract all parameter estimate from fit_AR2
phi1 = signif(coef(fit_AR2)[[1]],4)
phi2 = signif(coef(fit_AR2)[[2]],4)
mu = signif(coef(fit_AR2)[[3]],4)
sigma_z_square = signif(fit_AR2$sigma2,4)

# so the suggested model is
# (X_t-65.33)=0.5753*(X_t-1-65.33)-0.1501*(X_t-2-65.33)+Z_t;
# with var(Z_t) = 496.6

############## Model Diagnostic #############
# ASSUMING OUR MODEL IS CORRECT

# finally for diagnostic, plot the following:
# ACF plot: sample ACF with overlay theoretical ACF
# PACF plot: sample PACF with overlay theoretical PACF
# residual ts plot
# residual acf plot
# Ljung-Box test with plot
source("materials/Computer Practical 3 - R function for Ljung-Box test.R")
par(mfrow=c(2,1))

# sample ACF plot
acf(data, main="sample ACF plot", ylim=c(-1,1), lag.max=30)
# overlay theoretical ACF, using the parameter estimate
lines(0:30, ARMAacf(ar = c(phi1, phi2), ma = c(0), lag.max = 30), col="red")
# legend on top right
legend(x="topright",
       legend=c("sample ACF", "theoretical ACF"),
       col=c("black", "red"),
       lty=1)

# sample PACF plot
pacf(data, main="sample PACF plot", ylim=c(-1,1), lag.max=30)
# overlay theoretical PACF, using the parameter estimate
lines(1:30, ARMAacf(ar = c(phi1, phi2), ma = c(0), lag.max=30, pacf=TRUE), col="red")
# legend on top right
legend(x="topright",
       legend=c("sample PACF", "theoretical PACF"),
       col=c("black", "red"),
```

```
       lty=1)

par(mfrow=c(1,1))

# ACF and PACF are fitting quite good using our fitted model



## next see the residual plot and Ljung-Box test plot
resid_fit_AR2 = residuals(fit_AR2)

## using the material from Computer practical 3 for Ljung-Box test
# order of AR: p = 2
# order of MA: q = 0
# max.k = 12 so test degrees of freedom up to 10 (10 + p + q = 12)
LB_fit_AR2 = LB_test(resid_fit_AR2, max.k = 12, p = 2, q = 0)

par(mfrow=c(3,1))

# residuals time plot
plot(resid_fit_AR2, main="Residual Plot for the AR(2) fit", xlab="Time", ylab="Residual")

# residuals ACF plot
acf(resid_fit_AR2, main="ACF plot of residual for AR(2) fit", ylim=c(-1,1))

# LB-test result plotting
plot(LB_p_value ~ deg_freedom, data = LB_fit_AR2,
     main="Ljung-Box test: p-value vs degrees of freedom",
     xlab="degrees of freedom", ylab="p-value")

# add a reference line (horizontal 0.05)
abline(h=0.05, col="blue", lty=2)
par(mfrow=c(1,1))



# from the sample ACF plot, at lag 7, 14, 21
# there are ACF value lie outside the bound, but in fact they are close to the bound
# so it is insufficient to claim the residuals appears to have "seasonal" components
# from the Ljung-Box test plot,
# Because the LB test is used to test
# H0: residuals are independent vs H1: residuals are dependent
# we expect the p-value to be above 0.05 to retain H0
# However, from this model fit, p-values are below 0.05 on and after df=5
# it indicates the residuals are dependent


## next I define a helper function to plot these diagnostic plots
## then see the others fitted model
```

```
## data: the time series object
## fitted_model: the arima model
## p: order of AR component
## q: order of MA component
resid_diagnostic_plot = function(data, fitted_model, p, q){
  # extract the residuals of the model fit
  resid = residuals(fitted_model)

  # get the Ljung-Box test p-value for each degrees of freedom
  LB_fit = LB_test(resid, max.k = 10 + p + q, p = p, q = q)

  par(mfrow=c(3,1))

  # plot residuals plot
  plot(resid, main=paste("Residual Plot for the ARMA(",p,",",q,") fit"),
       xlab="Time", ylab="Residual")

  # plot residuals' ACF plot
  acf(resid, main=paste("ACF plot of residual for ARMA(",p,",",q,") fit"),
      ylim=c(-1,1))

  # plot Ljung-Box test results: p-value against df
  plot(LB_p_value ~ deg_freedom, data = LB_fit,
       main="Ljung-Box test: p-value vs degrees of freedom",
       xlab="degrees of freedom", ylab="p-value")
  # add a reference line: at 0.05 significance level
  abline(h=0.05, col="blue", lty=2)

  par(mfrow=c(1,1))

}

## my proposed model AR(2)
resid_diagnostic_plot(data, fit_AR2, p=2, q=0)

## rest of the fitted model
resid_diagnostic_plot(data, fit_AR1, p=1, q=0)
resid_diagnostic_plot(data, fit_AR3, p=3, q=0)
resid_diagnostic_plot(data, fit_MA1, p=0, q=1)
resid_diagnostic_plot(data, fit_MA2, p=0, q=2)
resid_diagnostic_plot(data, fit_ARMA11, p=1, q=1)
resid_diagnostic_plot(data, fit_ARMA12, p=1, q=2)
resid_diagnostic_plot(data, fit_ARMA21, p=2, q=1)

### none of these satisfied Ljung-Box test criteria
### so I may just end up with propose AR(2) to be the best model here
```

APPENDIX for Q2

```
######## Q2

###################### BASIC INVESTIGATION ########

rm(list=ls()) # remove environment variables

# set working directory
setwd("/Users/davidhui/Documents/GitHub/uon/MATH4022 Time Series and Forecasting/")

# read the eng_car_reg.csv
raw_data = read.csv("materials/eng_car_reg.csv")
head(raw_data)  # see the first few rows
n = nrow(raw_data)  # number of observations
n # n = 87

# convert the imported data into time series object
# frequency = 4 (quartly)
# with start at 1st quarter of 2001

data = ts(raw_data$no_new_regs, start=c(2001,1), frequency = 4)
data

# time series plot of the data
ts.plot(data, main="Quarterly number of new car registrations in England (in Thousands)",
        xlab = "Year",
        ylab = "numbers of new car registrations (in thousands)")

# it shows a seasonal pattern:
#   initially Q1 would be high,
#   then in Q2 would be a slight drop,
#   then in Q3 would rise again,
#   finally in Q4 would be a larger drop and reach the minimum of the year
# and repeats this pattern year by year generally

# it appears to have obvious outliers (in 2008 Q3, and 2020 Q2)
# in these 2 quarter, it has a large drop
# possible reasons: because 2008 Q3 had financial crisis;
# and 2020 Q2 had COVID-19 pandemic
# these events may have a large impact on new car registrations.

# except for where the outliers occurred,
# the variability are generally the same
# (no systematic change in variability)
# the mean is changing over different period.
# from 2001 to 2010, it appears to have a decreasing trend linearly
# from 2010 to 2017, it appears to have a increasing trend linearly
# finally from 2017 and afterwards, it showed a decreasing trend again
```

```
# to see the ACF and PACF plot
par(mfrow=c(2,1))
acf(data, main="ACF plot", lag.max=100)
pacf(data, main="PACF plot", lag.max=100)
par(mfrow=c(1,1))
# from the ACF plot, we can see that there are spikes for each lag multiples of 2
# to best describe this phenomenon,
# this is because from the data shows seasonal pattern
# so for every even quarters would have a drop;
# and for every odd quarters would have a rise
# so this will give a strong correlation for each lag multiples of 2
# however, we can see that the lag multiples of 4 would be a bit "larger"
# because of the influence of the seasonal period = 4


#################### MODEL IDENTIFICATION ####################


#### to start guessing the order of the SARIMA model ARIMA(p,0,q)x(P,1,Q)[4]
#### to start, because there are obvious seasonal trends, first remove seasonal pattern
data_diff4 = diff(data, lag=4)
layout(matrix(c(1,1,2,3), nc=2, byrow=T))
ts.plot(data_diff4, main="Time Series plot of seasonal differenced data (period=4)")
acf(data_diff4, ylim=c(-1,1), main="ACF plot of seasonal differenced data", lag.max=30)
pacf(data_diff4, ylim=c(-1,1), main="PACF plot of seasonal differenced data", lag.max=30)
par(mfrow=c(1,1))

# first from the time-series plot, even though there are some
# rapid fluctuation around 2010 and 2020,
# however apart from those that I determined as "outliers",
# we see that the rest parts of the plot looks "stationary"
# from the sample ACF and PACF plot,
# because both declines to 0 as lag increases

# we can see that both ACF and PACF plot, have a clear spike at lag 4 (seasonal period)
# except for the clear spike at lag 4,
# both ACF and PACF plot doesn't seem to have obvious cut-off
# it might show evidence that both AR and MA component exists in the model



################## MODEL FITTING ##################

## data = raw data
## p = non-seasonal AR order
## d = non-seasonal difference order
## q = non-seasonal MA order
## P = seasonal AR order
## D = seasonal difference order
```

```
## Q = seasonal MA order
## S = period
SARIMA_fit = function(data, p, d, q, P, D, Q, S){
  fit = arima(data, order=c(p,d,q), seasonal = list(order=c(P,D,Q), period=S),
              method="ML")
  print("===== fit summary =====")
  print(fit)
  print("===== test statistic =====")
  # coef(fit) returns the parameter estimation;
  # sqrt(diag(vcov(fit))) returns the s.e. of each parameter est.
  test_statistic = (coef(fit) / sqrt(diag(vcov(fit))))
  # no need to print "intercept
  print(test_statistic)

  # return the arima fit object, later I will need to extract the AIC
  return (fit)
}


df = as.numeric()

## I only try up to order 2
## and because I believe that the period is S = 4,
## only need perform 1st order of seasonal diff D = 1
## no need to further perform non-seasonal diff d = 0
## because arima(...) may though error if parameter out of specific range
## so there is another tryCatch block to handle this auto-search method
for(p in 0:2){
  for(q in 0:2){
    for(P in 0:2){
      for(Q in 0:2){
        print(paste("fitting: p=",p,"q=",q,",P=", P, ",Q=", Q))
        tryCatch({
          fit = SARIMA_fit(data, p, 0, q, P, 1, Q, 4)

          # check if this auto-search provides any error or warning or NaN
          # if so , ignore them
          test_statistic = (coef(fit) / sqrt(diag(vcov(fit))))
          if(sum(is.na(test_statistic)) == 0){

            ## we need the nonseasonal orders (p,q), seasonal orders (P,Q),
            tmp = data.frame(p = p, q = q, P = P, Q= Q, AIC = AIC(fit))
            df = rbind(df, tmp)
          }
        }, error = function(err){
          message(err)
          print(paste("error in fitting: p=",p,"q=",q,",P=", P, ",Q=", Q))
        }, warning = function(warn){
          message(warn)
          print(paste("warning in fitting: p=",p,"q=",q,",P=", P, ",Q=", Q))
```

```
        })
      }
    }
  }
}

# see the result
df

library("dplyr")

# sort by AIC
df %>% arrange(AIC)
# from this greedy auto-search, we can find that
# there are 76 out of 81 iterations give valid AIC
# we can checkout the rest of them

# see the first 10 only
df %>% arrange(AIC) %>% head(10)


# choose ARIMA(1,0,1)(0,1,1)[4]
fitted_model = SARIMA_fit(data, p=1, d=0, q=1, P=0, D=1, Q=1, S=4)


################### MODEL DIAGNOSTIC ####################
### 1. ACF and PACF with overlay theoretical ACF/PACF (on seasonal diff data)
### 2. residuals plot
### 3. residuals ACF plot
### 4. Ljung-Box test


## 1. ACF and PACF with overlay theoretical ACF/PACF
# using our model ARIMA(1,0,1)(0,1,1)[4]
# can be read as ARMA(1,5) for the seasonal differenced data

phi1 = 0.9845;
theta1 = -0.5401;
Theta1 = -0.9360;
S = 4;
ar = c(phi1)
ma = c(theta1, 0, 0, Theta1, theta1*Theta1)

# sample ACF plot on the seasonal differenced data
acf(data_diff4, main="sample ACF plot of seasonal diff data",
    ylim=c(-1,1), lag.max=30)
# overlay theoretical ACF, using the parameter estimate
# note that we need the "Lag" divided by the seasonal period
lines((0:30)/S, ARMAacf(ar = ar, ma=ma, lag.max = 30), col="red")
# legend on top right
```

```
legend(x="topright",
       legend=c("sample ACF", "theoretical ACF"),
       col=c("black", "red"),
       lty=1)

# sample PACF plot on the seasonal differenced data
pacf(data_diff4, main="sample PACF plot of seasonal diff data"
     , ylim=c(-1,1), lag.max=30)
# overlay theoretical PACF, using the parameter estimate
lines((1:30)/S, ARMAacf(ar = ar, ma = ma, lag.max = 30, pacf=TRUE)
      , col="red")
# legend on top right
legend(x="topright",
       legend=c("sample PACF", "theoretical PACF"),
       col=c("black", "red"),
       lty=1)

# ACF and PACF are fitting quite good using our fitted model

### residuals plot
fitted_model_resid = residuals(fitted_model)
ts.plot(fitted_model_resid,
        main="Residuals plot")

### residuals ACF plot
acf(fitted_model_resid,
    main="Residuals ACF plot")


### Ljung-Box test
source("materials/Computer Practical 3 - R function for Ljung-Box test, SARIMA MODEL.R")
## using the material from Computer practical 3 for Ljung-Box test
# non-seasonal order of AR: p = 1
# non-seasonal order of MA: q = 1
# seasonal order of AR: P = 0
# seasonal order of MA: Q = 1
# max.k = 13 so test degrees of freedom up to 10 (10 + p + q + P + Q= 13)
LB_fitted_model_resid = LB_test_SARIMA(fitted_model_resid, max.k=13, p=1, q=1, P=0, Q=1)

# plot the Ljung-Box test resulting plot
plot(LB_p_value ~ deg_freedom, data = LB_fitted_model_resid,
     main="Ljung-Box test: p-value vs degrees of freedom",
     xlab="degrees of freedom", ylab="p-value",
     ylim=c(0,1))

# add a reference line (horizontal 0.05)
abline(h=0.05, col="blue", lty=2)


################## FORECASTING #################
```

```
library("forecast")
## using forecast method
## data = raw data
## model = using our ARIMA(1,0,1)(0,1,1)[4]
## h = 4, our required future time step for forecasting
## level = 95, the confidence level
forecast_value = forecast(data, model=fitted_model, h=4, level=95)


# check out the forecasted value with uncertainty
# uncertainty = (upper95% bound - lower95% bound ) / 2
fc = forecast_value %>%
  as.data.frame(.) %>%
  rename("Lower" = "Lo 95", "Higher" = "Hi 95") %>%
  mutate("Uncertainty" = (Higher - Lower)/2)

fc


# combine raw data with forecasting mean and uncertainty
data_with_fc = cbind(forecast_value$mean,
                     forecast_value$upper,
                     forecast_value$lower,
                     data)

# plot those data
# together with associated uncertainty
autoplot(data_with_fc,
         main="Quarterly number of new cars registered with prediction",
         xlab="Year", ylab="number of new cars registeration (in thousands)")
```