Analytical Approach: After loading the dataset provided by the course instructor, I have considered V48(pre-assigned) as a time series data, which has 100 observations with no seasonal components. Plotting and testing several tests for stationarity, second order differencing makes the data stationary. After that, successfully determining p_{max} and q_{max} , I have made models with ARIMA approach for all possible combinations of p and q and checked the AIC (minimum), BIC (minimum), and diagnostics of models to conclude a final model that I want go further. Now, the final model is concluded, it's time to forecast and observe the plots.

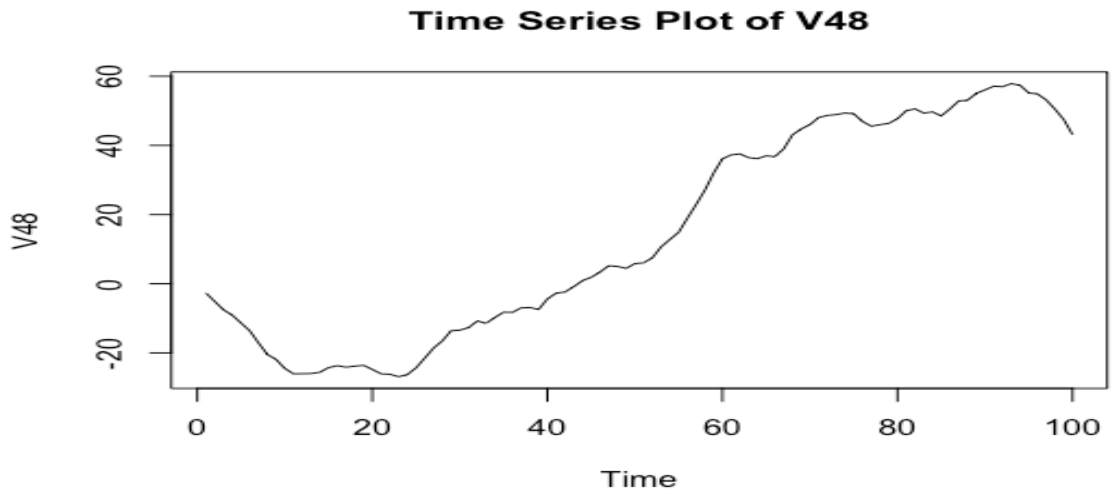
Step 1: Preliminary analysis of Orders

```
# Checking the working directory

getwd()

# Loading the necessary libraries to conduct time series analysis

library(tseries)

library(urca)
```

```
library(forecast)

library(TSA)

library(readr)

# Importing the dataset

Case_study = read_csv("Case_study.csv")



# Specifying my assigned column V48 as time series data

data = ts(Case_study$V48)

#Checking whether the dats is time series data ot not

class(data)

time(data)

season(data) # there is no seasonality.
```

1. Plotting original time series data

```
# Plotting the time series data

plot(data, main = "Time Series Plot of V48", xlab = "Time", ylab = "V48")
```



This time series is not stationary by visual representation

It is showing an increasing trend and a non-constant variance

Augmented Dickey–Fuller test

adf.test(data, alternative = "stationary")

H0 : Data is not stationary

p-value = 0.7914

High p value, H0 is not rejected

Need to differencing for order until H0 gets rejected

2. Analysis of d:

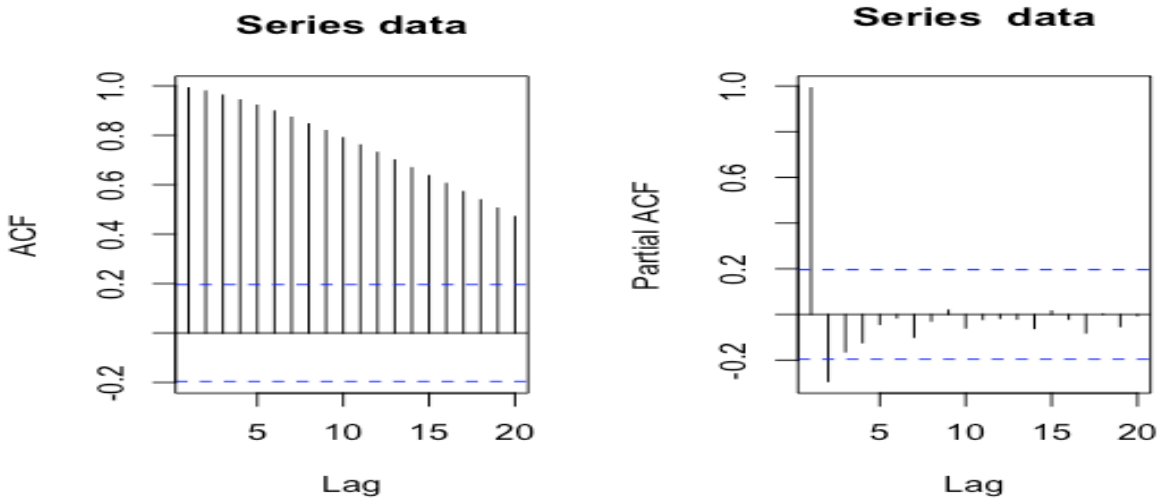
Autocorrelation and partial autocorrelation at different lags

par(mfrow = c(1,2))

acf(data) # acf is dampening as the lags increases.

pacf(data) # pacf is zero after lag 1

par(mfrow = c(1,1))



the recommended model is AR(1) model

blue is the significant zero line

Significant zeroes are after lag 1

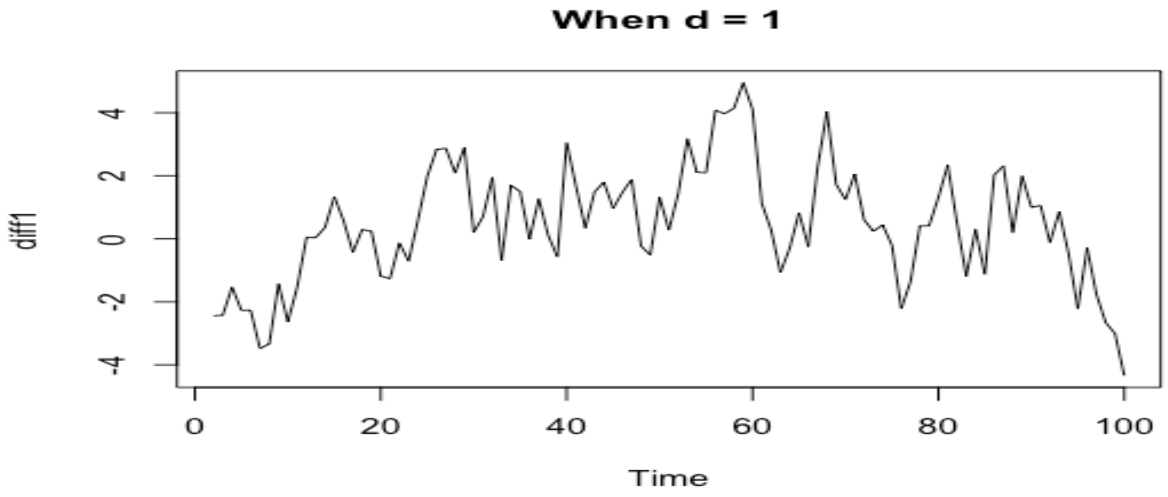
So the approach will be AR(1)

To make the data stationary, I am going to difference the ts data

diff1 = diff(data, differences = 1) ; diff1

class(diff1)

plot(diff1, main = "When d = 1")



Showing non constant variance and formed a slight curvier shape

For Augmented Dicky Fuller test,

Null Hypothesis: Data series is not stationary.

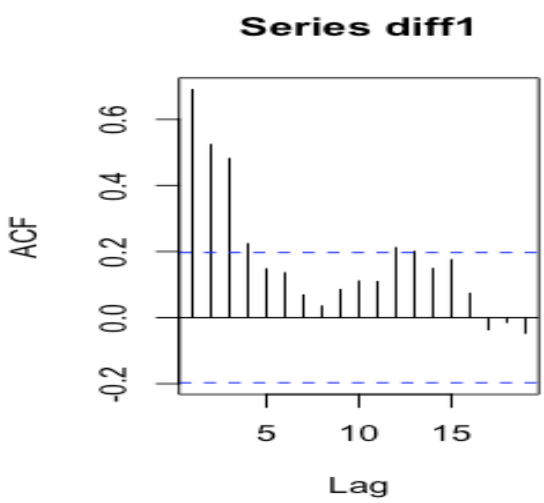
Alternative hypothesis: Data series is stationary.

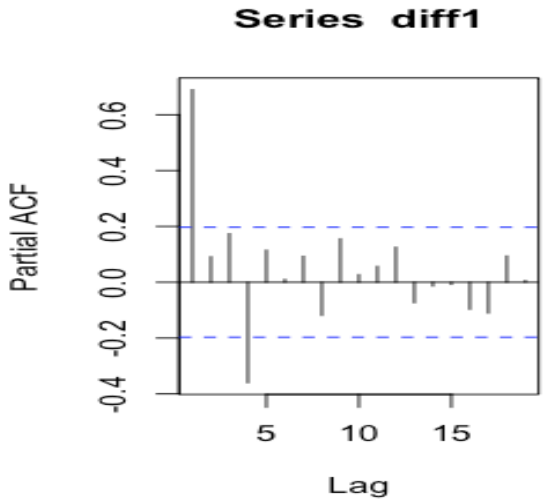
par(mfrow = c(1,2))

acf(diff1)

pacf(diff1)

par(mfrow = c(1,1))





adf.test(diff1)$p.value

not stationary at k = 1 i.e at first order

Since the p value of ADF test is around 0.4 which is much greater than 5% level of significance, we cannot reject null hypothesis of being not stationary. Thus, first order of time series data is not stationary.

Now, its time to test the second order differencing

diff2 = diff(data, differences = 2) ; diff2

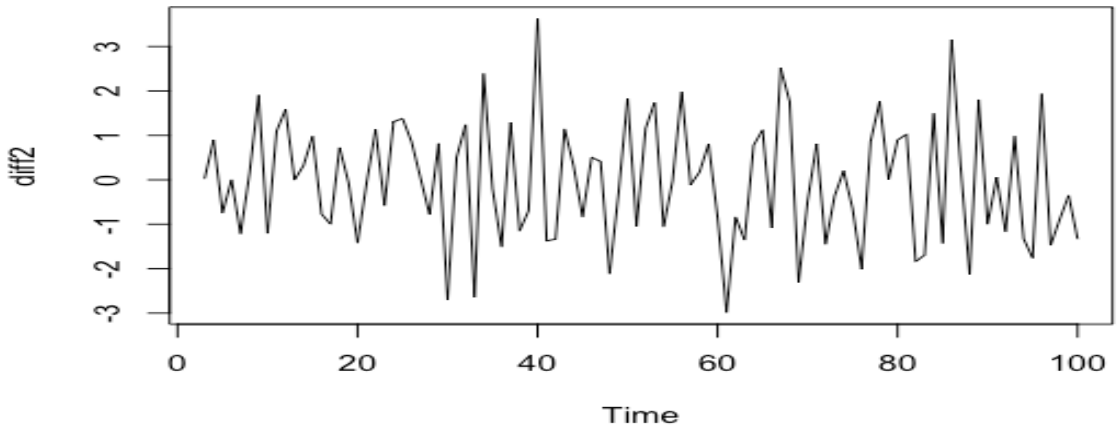
plot(diff2)

par(mfrow = c(1,2))

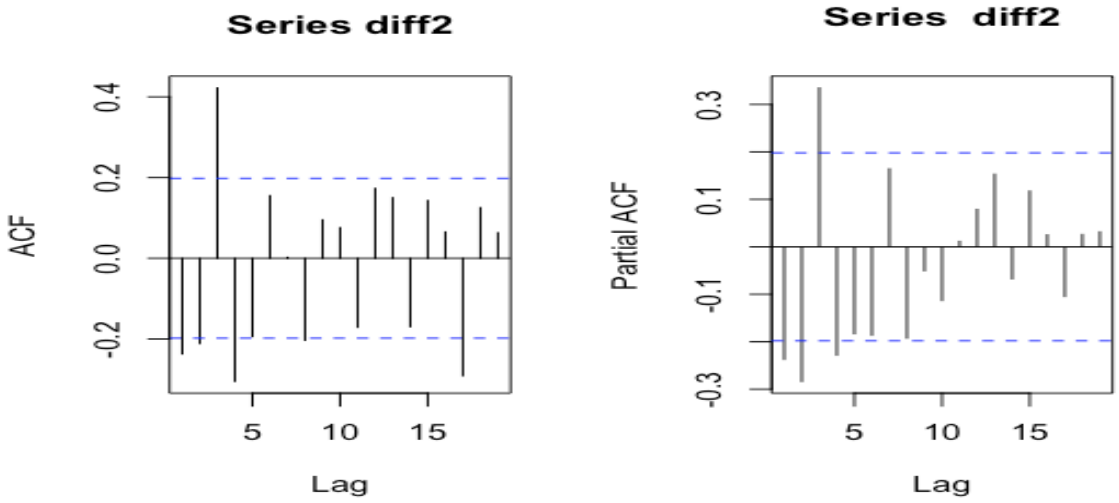
acf(diff2)

pacf(diff2)

par(mfrow = c(1,1))



Plot of Second Order Differencing



ACF and PACF Plot

adf.test(diff2) # p = 0.01 Stationary at k = 2

Since the p value of ADF test for second order difference is around 0.01 which is much smaller than 5% level of significance. This time, we can reject null hypothesis of being not stationary. Thus, second order of time series data is stationary.

When we go with ARIMA approach, we can put d=2

3. Analysis of p and q

Upon observing the ACFs and PACFs we can conclude that upper bound for both p and q could be 4 because in both ACF and PACF of second order time series, all values fallen to zero significant blue boundary after lag 4.

Step 2: Estimation and Selection of ARIMA Models

1. Estimation of ARIMA

```
# Define the Upper bound of p and q

pmax = 4

qmax = 4



# Create an empty dataframe to store AIC and BIC values

AIC_BIC = data.frame(p = integer(0), q = integer(0), AIC = numeric(0), BIC
= numeric(0))



# Loop through all combinations of p and q

for (p in 1:pmax) {

  for (q in 1:qmax) {

    # Skip combinations (0, 0), (0, q), and (p, 0)

    if (p == 0 && q == 0) next

    if (p == 0 || q == 0) next
```

```
    # Fit ARIMA model
    model = arima(data, order = c(p, 2, q))

    # Compute AIC, BIC
    aic = AIC(model)
    bic = BIC(model)

    # Store results in the dataframe
    AIC_BIC = rbind(AIC_BIC, data.frame(p = p, q = q, AIC = aic, BIC = bic))
  }
}

# Printing the dataframe with AIC and BIC values
print(AIC_BIC)
```

Index	p	q	AIC	BIC
1	1	1	332.6153	340.3702
2	1	2	322.2971	332.6370
3	1	3	321.3840	334.3089
4	1	4	309.1300	324.6398
5	2	1	315.7763	326.1162
6	2	2	317.4581	330.3829
7	2	3	313.6365	329.1463
8	2	4	310.0283	328.1231
9	3	1	317.2358	330.1606
10	3	2	319.0883	334.5981
11	3	3	313.7297	331.8244
12	3	4	309.8291	330.5088
13	4	1	312.1255	327.6353
14	4	2	314.1080	332.2028
15	4	3	314.6167	335.2965
16	4	4	311.7655	335.0302

I have set the p max and q max as 4, and I have found that second order differencing is the stationary process. Therefore, I have written an empty data frame for p, q, AIC, and BIC and in a loop I have fitted the ARIMA model for all the possible combinations of p and q(with respect to skipping the combinations (0,0), (p,0), and (q,0))

2. Best specifications

Accoding the AIC and BIC of the all possible combinations

```
which.min(AIC_BIC,"AIC"])  #4

which.min(AIC_BIC,"BIC"])  #4

# Both showing index 4

AIC_BIC[4,]
```

Index	p	q	AIC	BIC
4	1	4	309.1300	324.6398

Which is suggesting ARIMA(1,2,4)

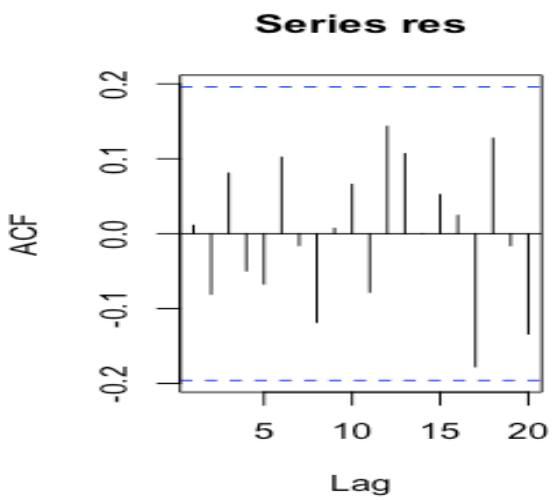
But with the auto arima approach it is suggested that, we can also fit the model with ARIMA(2,2,1) approach, since This model provides second lowest BIC.
Now we have two models
ARIMA(1,2,4) and ARIMA(2,2,1)

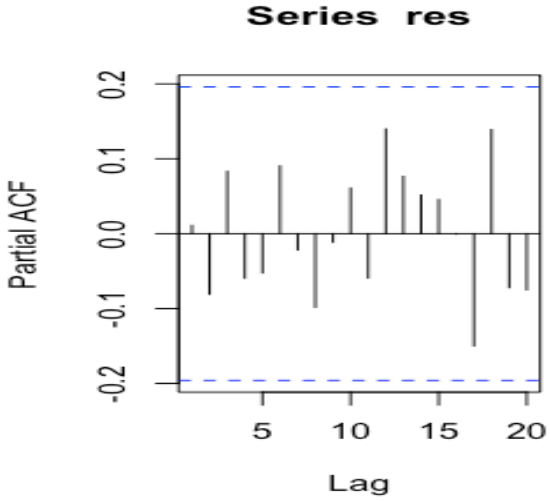
Step 3: Diagnostic tests

1. LjungBox test, ACF and PACF of the Two models

For, ARIMA(1,2,4)

```
model_1 = arima(data, order = c(1,2,4))

res = resid(model_1)

# H0: Data is not stationary

adf.test(res)

# since p = 0.01, res is stationary

par(mfrow = c(1,2))

acf(res)

pacf(res)
```





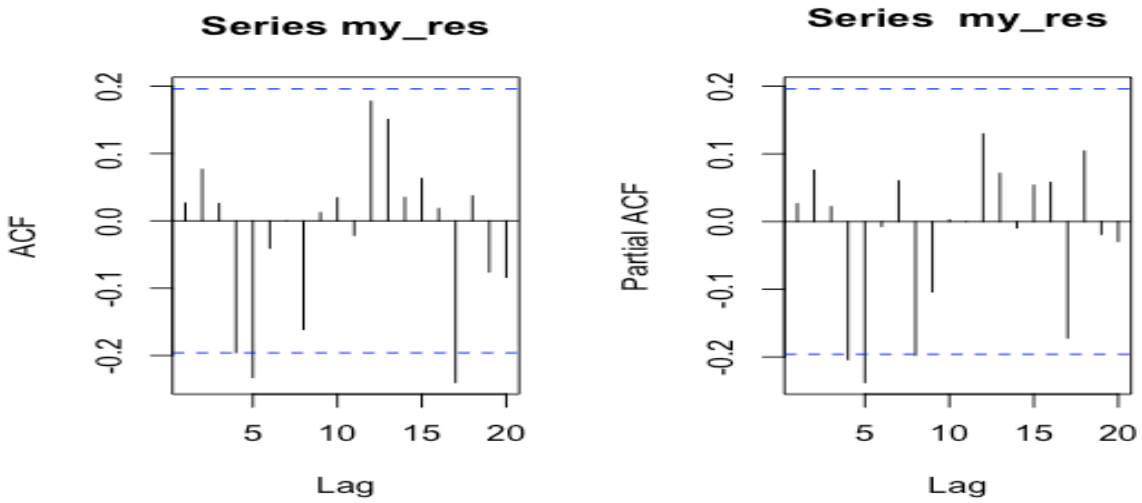
```
par(mfrow = c(1,1))
```

In both ACF and PACF it is evident that there is no autocorrelation

```
# H0 : There is no autocorrelation in the residuals

Box.test(res, lag = 10, type = "Ljung-Box")

# We can not reject null hypothesis

# There is no correlation in the residuals in the model
```

Now, For ARIMA(2,2,1)

```
### auto arima and arima(2,2,1) is same thing

### Second lowest BIC

my_arima = auto.arima(data) ; my_arima

my_res = resid(my_arima) ; my_res

# H0: Data is not stationary

adf.test(my_res)

# since p = 0.01, my_res is stationary

par(mfrow = c(1,2))

acf(my_res)

pacf(my_res)
```



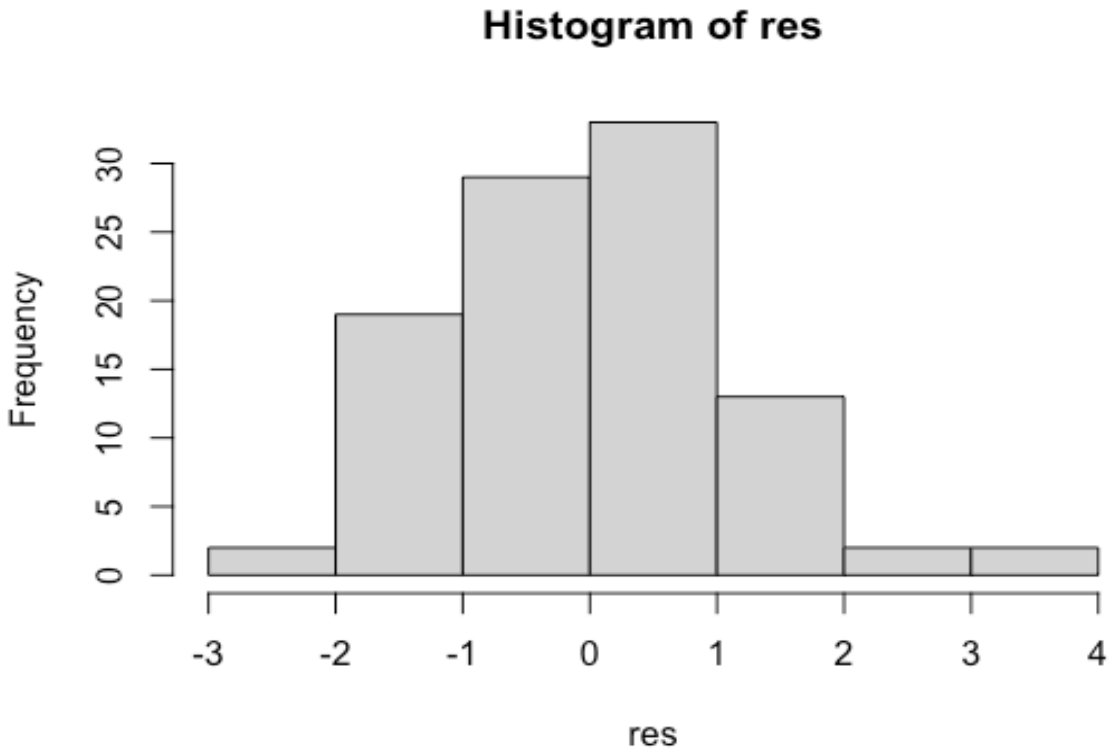

```
par(mfrow = c(1,1))

# H0 : There is no autocorrelation in the residuals

Box.test(my_res, lag = 10, type = "Ljung-Box")

# p value is 0.18

# There is no autocorrelation
```

According to LjungBox test, residuals from ARIMA(2,2,1) are not autocorrelated. It can also be determined ny the acf and pacf graphs. Though at some lag it is over the significant boundary but the amount is quite small.

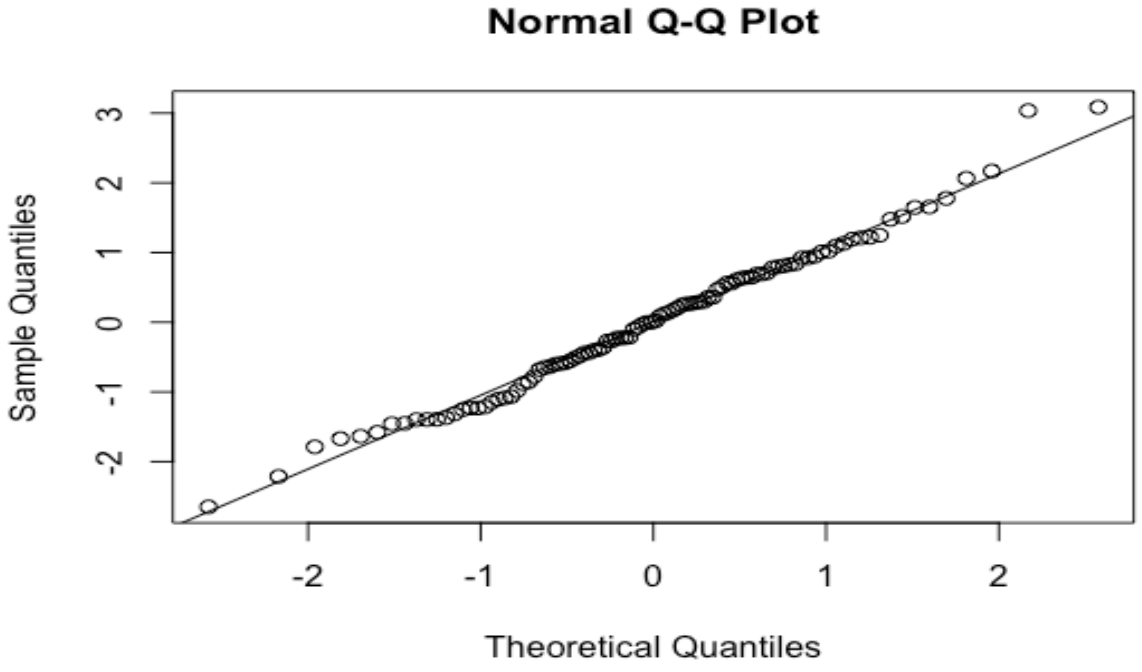
2. Normality test for residuals for both model

ARIMA(1,2,4)

```
# Normality test
hist(res)
```



qqnorm(res)
qqline(res)



H0: Observations are normally distributed

shapiro.test(res)

P value is 0.6096 which means that we can not reject null hypothesis, residuals are normally distributed.

We can also test normality by testing skewness and kurtosis

Skewness and Kurtosis observation

library(moments)

skew = skewness(res) ; skew

0.2256767 (For normal distribution, skew = 0 is ideal value)

Though skew is 0.22, slightly positive skewed ,but looks like normal, its not a big deal

kurt = kurtosis(res) ; kurt

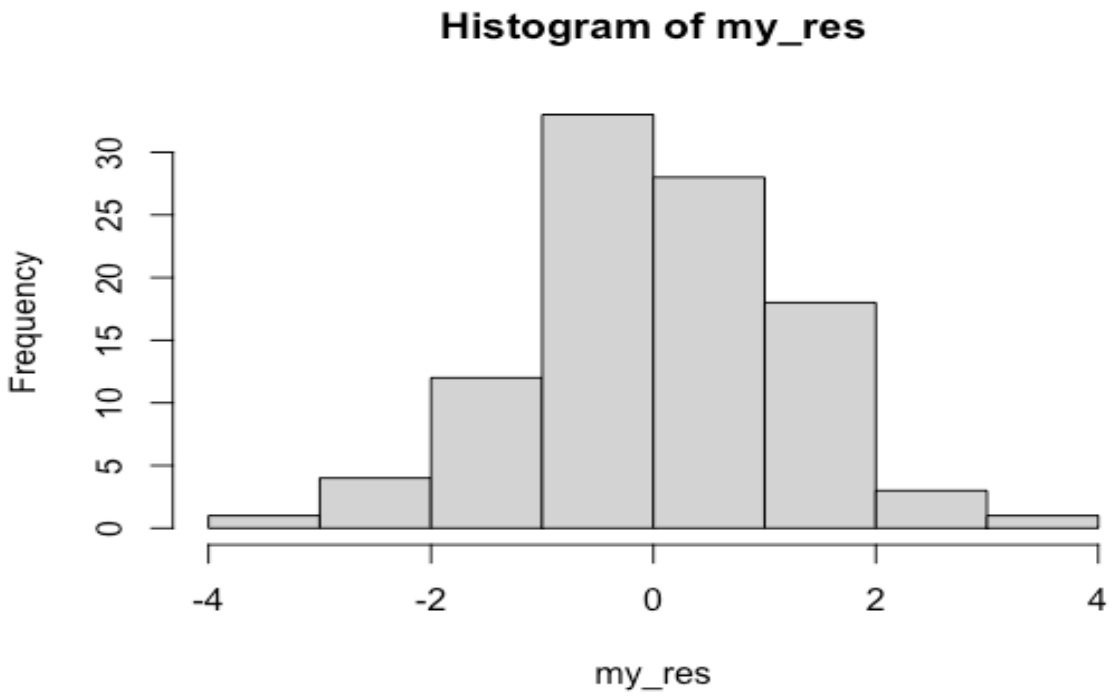
3.056607 (For normal distribution, kurt = 3 is ideal value)

Since kurtosis coefficient is 3, peakwise res seems like normal.

For ARIMA(2,2,1)

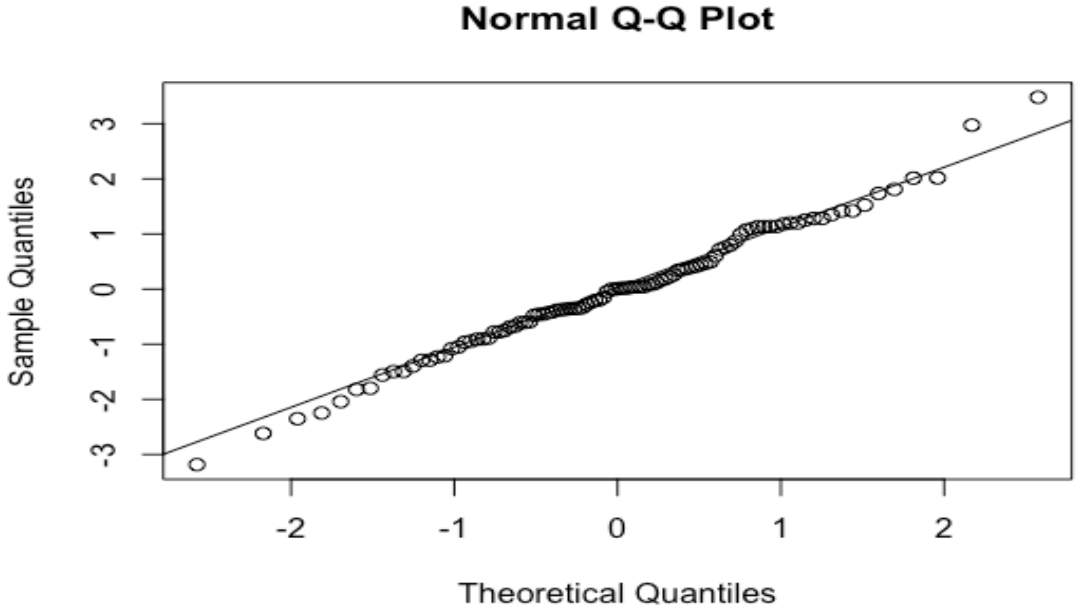
testing for arima(2,2,1)

hist(my_res)



qqnorm(my_res)

qqline(my_res)



shapiro.test(my_res)

P value is 0.704, which means that residuals are normally distributed

skewness(my_res)

0.08431102 (Perfectly normal)

kurtosis(my_res)

3.497166 (Though it is greater than 3.0, But looks like normal, this kind value is accepted while sample size is large,)

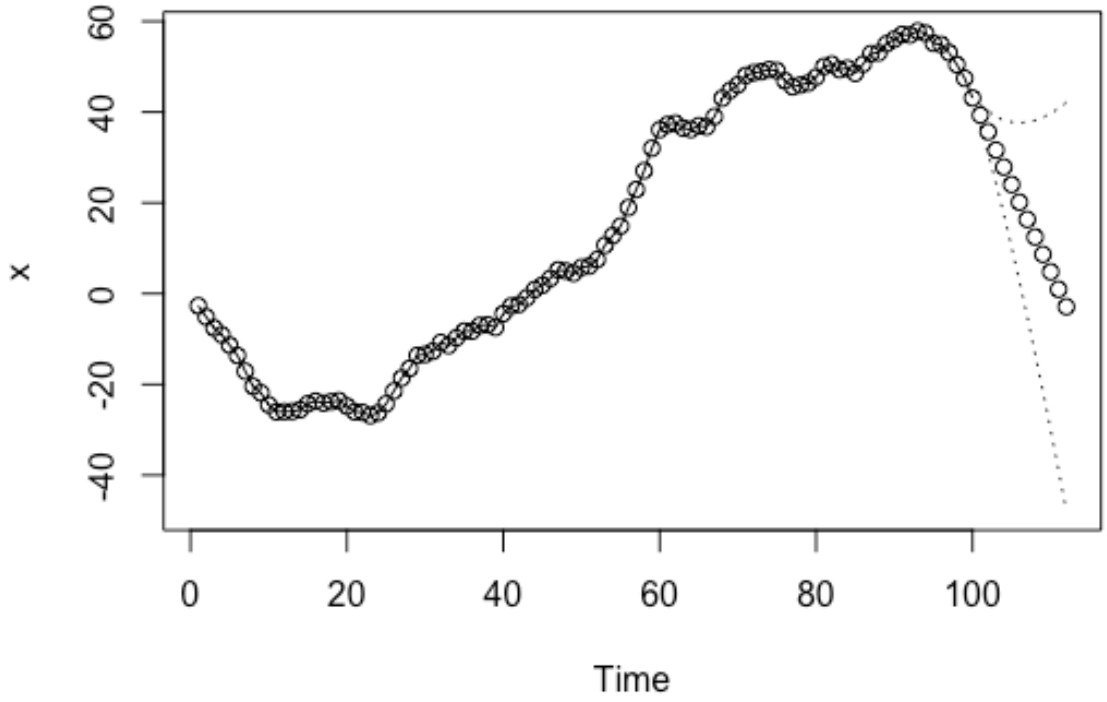
So, Both ARIMA models residuals are normal. But visual wise, ARIMA(2,2,1) is slightly strong towards normality.

3. Model preference

After studying all the factors and the suggestions by the course instructor (among two competing models, one would generally select the one with less parameters, especially with respect to the moving average component), I want to proceed with ARIMA(2,2,1) model.

4. Plotting original time series and best fitted model (2,2,1)

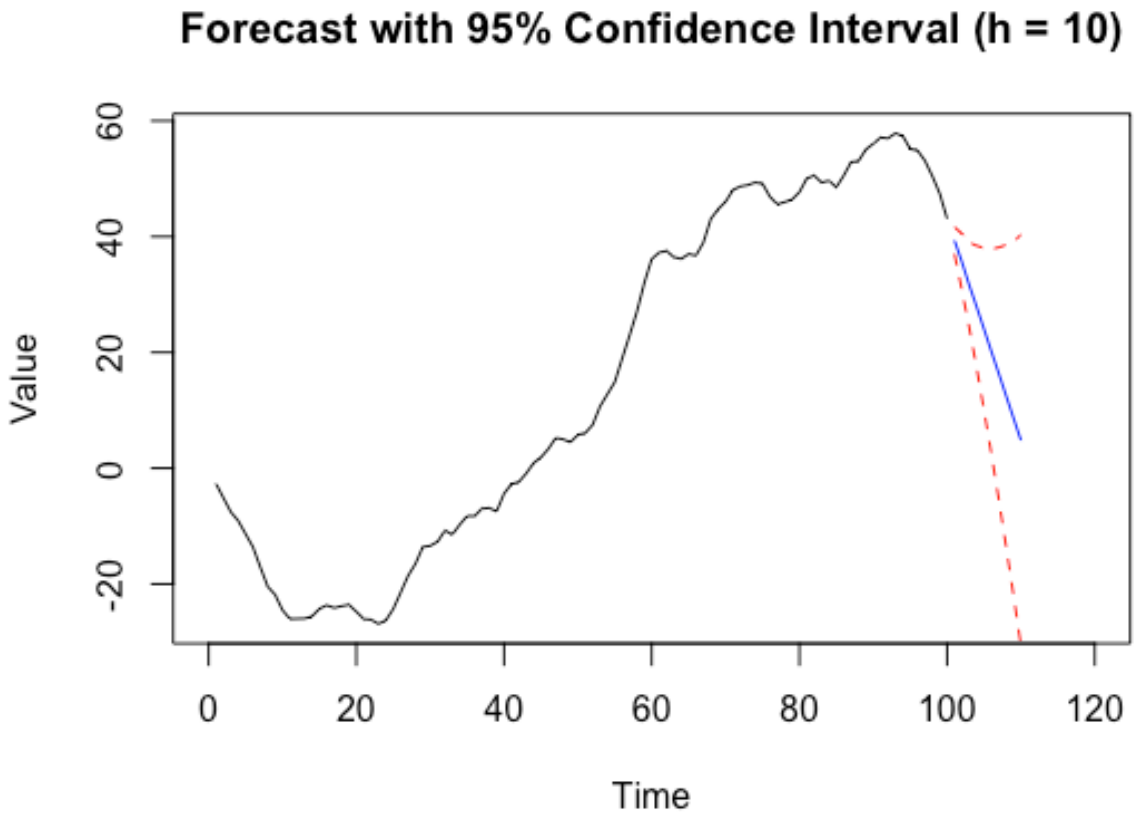
```
#       Plotting original time series and best fitted model (2,2,1)

plot(arima(data, order = c(2,2,1)))
```



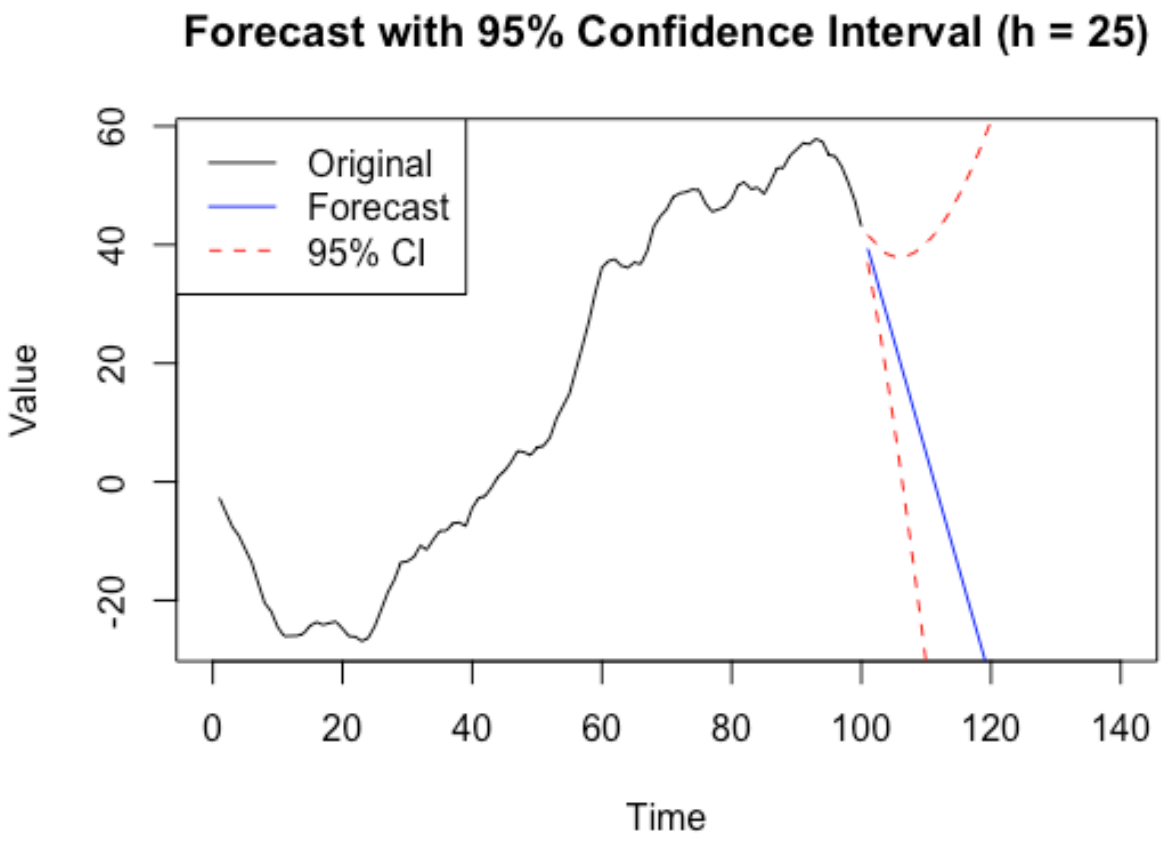
Plot of the original time series and fitted model(2,2,1)

Step 4: Forecast

```
# forecasting

my_arima = auto.arima(data)

my_arima

# When h=10 and level=95

forecast_10 = forecast(my_arima, h = 10, level = 95) ; forecast_10

#When h=25 and level=95

forecast_25 = forecast(my_arima, h = 25, level = 95) ; forecast_25

# Plotting the original time series with forecasts and confidence intervals
(10,95)
plot(data, main = "Forecast with 95% Confidence Interval (h = 10)",
     xlab = "Time", ylab = "Value", xlim = c(0,120))
lines(forecast_10$mean, col = "blue")
lines(forecast_10$lower, col = "red", lty = 2)
lines(forecast_10$upper, col = "red", lty = 2)
legend("topleft", legend = c("Original", "Forecast", "95% CI"), col =
c("black", "blue", "red"), lty = c(1, 1, 2))
```




```
# Plot the original time series with forecasts and confidence intervals for h
= 25
plot(data, main = "Forecast with 95% Confidence Interval (h = 25)",
     xlab = "Time", ylab = "Value", xlim = c(0,140))
lines(forecast_25$mean, col = "blue")
lines(forecast_25$lower, col = "red", lty = 2)
lines(forecast_25$upper, col = "red", lty = 2)

# Add legend
legend("topleft", legend = c("Original", "Forecast", "95% CI"), col =
c("black", "blue", "red"), lty = c(1, 1, 2))
```


Forecast with 95% Confidence Interval (h = 25)
Original
Forecast
95% CI
Value
60
40
20
0
-20
0
20
40
60
80
100
120
140
Time