EE798 FISA Assignment

Sandeep Parmar

210922

Introduction

India's coal production is predominantly reliant on open-pit mining, with a projected surge in future coal demand. However, the significant environmental challenges arising from this mining method, particularly the deterioration of air quality due to the release of particulate matter and various gaseous pollutants, present substantial constraints on coal utilization. Assigned by Dr. Tushar Sandhan, I undertook an in-depth analysis of pollutant data, unearthing invaluable insights that hold the potential to mitigate air pollution and foster a healthier environment.

Original Data

The air pollution data set is obtained from the Singrauli Coalfield Pollution Control Board for coal India's (Singrauli Coalfield). The pollution is monitored during open-pit blasting.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	#	From	To (Interval: 15M)	Singrauli, S	Singrauli, S	Singrauli, S	Singrauli, S	Singrauli, S	Singrauli, S	Singrauli, S	Singrauli, Su	Singrauli, S	Singrauli, S
2	1	01-02-2023 00:00	01-02-2023 00:15	95	35	NA	90.1	56.2	0.31	NA	17.7	28.1	0.4
3	2	01-02-2023 00:15	01-02-2023 00:30	95	35	NA	88	55.1	0.33	NA	18.3	27.1	0.4
4	3	01-02-2023 00:30	01-02-2023 00:45	95	35	NA	87.7	55.2	0.38	NA	19.7	24.9	0.4
5	4	01-02-2023 00:45	01-02-2023 01:00	122	34	NA	88.9	55.7	0.38	NA	21.3	21.9	0.4
6	5	01-02-2023 01:00	01-02-2023 01:15	122	34	NA	90	55.8	0.38	NA	22.3	16.7	0.4
7	6	01-02-2023 01:15	01-02-2023 01:30	122	34	NA	90.2	55.9	0.37	NA	22.7	16.1	0.4
8	7	01-02-2023 01:30	01-02-2023 01:45	122	34	NA	88.9	55.4	0.34	NA	23.1	22.5	0.4
9	8	01-02-2023 01:45	01-02-2023 02:00	90	35	NA	88.9	55.2	0.35	NA	23.5	20.5	0.4
10	9	01-02-2023 02:00	01-02-2023 02:15	90	35	NA	88.9	55.9	0.34	NA	23.1	22.8	0.4
11	10	01-02-2023 02:15	01-02-2023 02:30	90	35	NA	88.9	55.3	0.35	NA	22.9	19	0.4
12	11	01-02-2023 02:30	01-02-2023 02:45	90	35	NA	88.8	55.4	0.39	NA	22.6	17.1	0.4
13	12	01-02-2023 02:45	01-02-2023 03:00	72	32	NA	88.9	55.1	0.46	NA	22.5	19.1	0.4
14	13	01-02-2023 03:00	01-02-2023 03:15	72	32	NA	88.9	55.8	0.47	NA	22.7	18.2	0.4
15	14	01-02-2023 03:15	01-02-2023 03:30	72	32	NA	87.7	55.5	0.44	NA	22.5	18	0.4

Figure 1: Original Data Snapshot

The air pollution data collection comprises a total of 13 columns, containing information on various pollutants. These data points are recorded at 15-minute intervals, resulting in a grand total of 8,640 observations spanning from 01-02-2023 00:00:00 IST to 01-05-2023 23:45:00 IST, covering a period of 90 days. The concentration of pollutant particles is measured in $\hat{A}\mu/m^3$.

Description of Columns

- column (A) Indicates serial no of the data set.
- column (B) and (C) Indicates date and time from and to for 15 minutes of interval.
- column (D) Indicates PM10 pollutant of the data.

- column (E) Indicates PM2.5 pollutant of the data.
- column (F) Indicates NO pollutant of the data.
- column (G) Indicates NO2 pollutant of the data.
- column (H) Indicates NOX of the data.
- column (I) Indicates CO pollutant of the data.
- column (J) Indicates SO2 pollutant of the data.
- column (K) Indicates NH3 pollutant of the data.
- column (L) Indicates Ozone pollutant of the data.
- column (M) Indicates Benzene pollutant of the data.

Problem Statement

1. What type of probability distribution do the pollutants follow, and how can we test it using hypothesis testing?
2. Detect Blasting Time from data.
3. Predict the concentration through Time-Series Model.

Pre-processing of Data

I used R programming language in RStudio framework for all purposes in the analysis. Below is the main changes that I have made in parent data.

- Changing of column name.
- Remove last extra 3 row of parent data which contain summary of that column.
- Convert string of date & time to Standard Time format.
- Add extra columns of Hour, Day and Month.

Code :-

```
#load the dataset
setwd("C:/Users/psand/OneDrive/Desktop/EE798_FISA/Assignment_1")
df <- read.csv("Open pit blasting 01-02-2023 000000 To 01-05-2023 235959.csv")

# Changing column name
colnames(df) <- c("s_num","From","To","PM10","PM2.5","NO",
                  "NO2","NOX","CO","SO2","NH3","ozone","benzene")

# Standard Time format
df$From <- as.POSIXct(df$From,format="%Y-%m-%d %H:%M:%OS")
df$To   <- as.POSIXct(df$To,format="%Y-%m-%d %H:%M:%OS")

# Removing last rows
df <- df[-c(8641,8642,8643),]
```

```r
# Extra Column of month,day,hour
df$month <- as.numeric(format(df$From,"%m"))
df$day <- as.numeric(format(df$From,"%d"))

# For hour
str <- df$From
str <- as.character(str)
sstr<-strsplit(str,split = " ")
hr <- array(data = NA,dim = 8640)
for(i in 1:8640){
  hr[i] <- sstr[[i]][2]
}
shr <- strsplit(hr,split = ":",fixed = T)
for(i in 1:8640){
  hr[i] <- shr[[i]][1]
}
df$hr <- as.numeric(hr)
```

Descriptive Statistics

To conduct a descriptive analysis, I utilized the `summary()` function in the R and generate a dataset.

	Particle	Min_Value	Qunt_1	Median	Mean	Qunt_3	Max_value	NAs	std.dev	uniq_entry
1	PM10	12.0	84.00	145.00	181.500	238.00	847.0	1681	136.01	376
2	PM2.5	3.0	36.00	61.00	75.730	101.00	474.0	226	55.24	245
3	NO	0.1	3.90	6.10	14.670	16.50	157.5	1396	19.22	775
4	NO2	0.2	39.40	53.20	55.760	71.05	106.9	416	20.23	861
5	NOX	4.2	25.00	37.70	42.680	53.80	165.9	415	22.43	1009
6	CO	0.1	0.95	1.42	1.409	1.85	4.0	496	0.63	298
7	SO2	0.1	16.10	25.30	34.310	35.20	645.6	1451	39.45	1113
8	NH3	4.6	9.40	11.00	13.250	14.00	62.4	326	6.15	334
9	Ozone	0.1	10.50	32.40	35.630	58.80	123.8	453	27.01	973
10	Benzene	0.1	0.10	0.10	0.178	0.20	0.6	6195	0.09	7

Key points :-

- The concentration of PM10 particles is the highest, while the concentration of Benzene particles is the lowest.
- Benzene particles have the highest number of NA values, and there are only seven unique entries for Benzene in the given dataset. The unique entries are [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, NA].
- SO2 and PM10 particles have roughly equal NA values, but the standard deviation of PM10 is much higher than that of SO2. Interestingly, SO2 has the highest number of unique observations, while PM10 has very few.
- PM2.5 has the minimum number of NA values, making it useful for obtaining a general overview of the entire dataset, including trends and seasonality.

Interpolation

Since there are numerous missing entries in the parent data, it is essential to address the NA values in each column to conduct a comprehensive analysis. In this case, I used *Cubical Spline Interpolation* to fill in the missing values.

For that I used `na.spline()` function from `zoo` package of R.

code:-

```
library(zoo)
df[ ,c(4:14)] <- na.spline(df[ ,c(4:14)])
```

During the analysis, I noticed that the NA values in the NO and SO2 columns were filled using negative values. This indicates that cubic spline interpolation was not suitable for filling those columns missing values.

Lets see how NA values is filled in PM10 time-series graph .

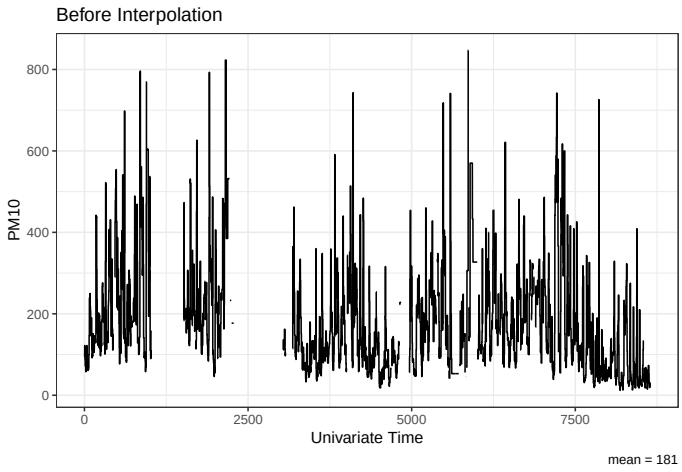


Figure 2: (2)

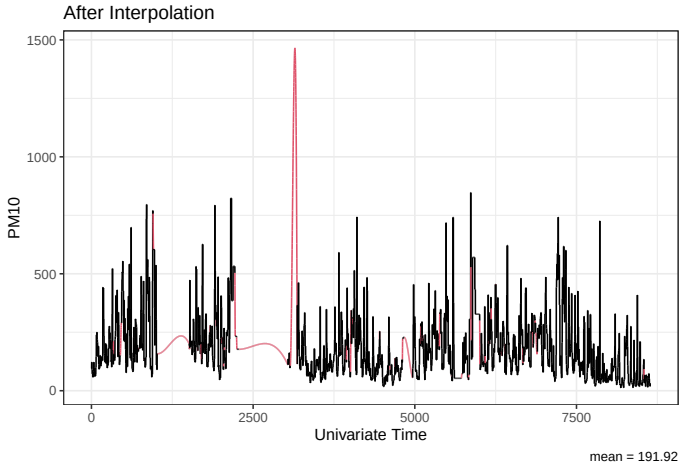


Figure 3: (2)

Here we can see how nicely curve is connected and there is not too much change in mean concentration of PM10.

Note:- In all the plots, the univariate time series represents time according to the data, ranging from 1 to 8640. For instance, 1 corresponds to 01/02/2023 00:00:00 IST, and 2 represents 01/02/2023 00:15:00 IST, with a frequency of 15 minutes. The final time, 8640, represents 01/05/2023 23:45:00 IST.

Exploratory analysis

First have a look time series plot of all pollutant across whole univariate time series.

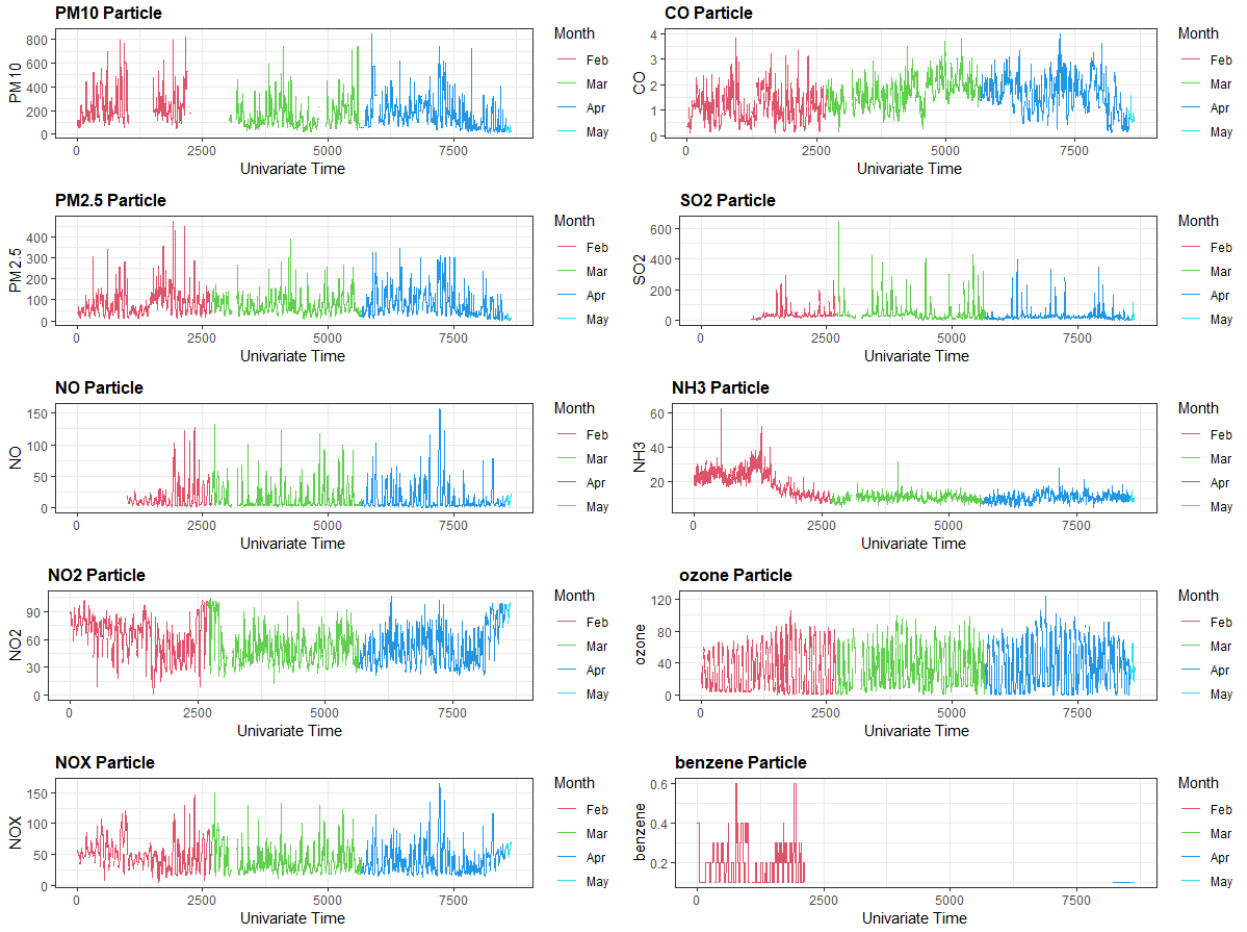


Figure 4: All pollutant time-series plot of original data

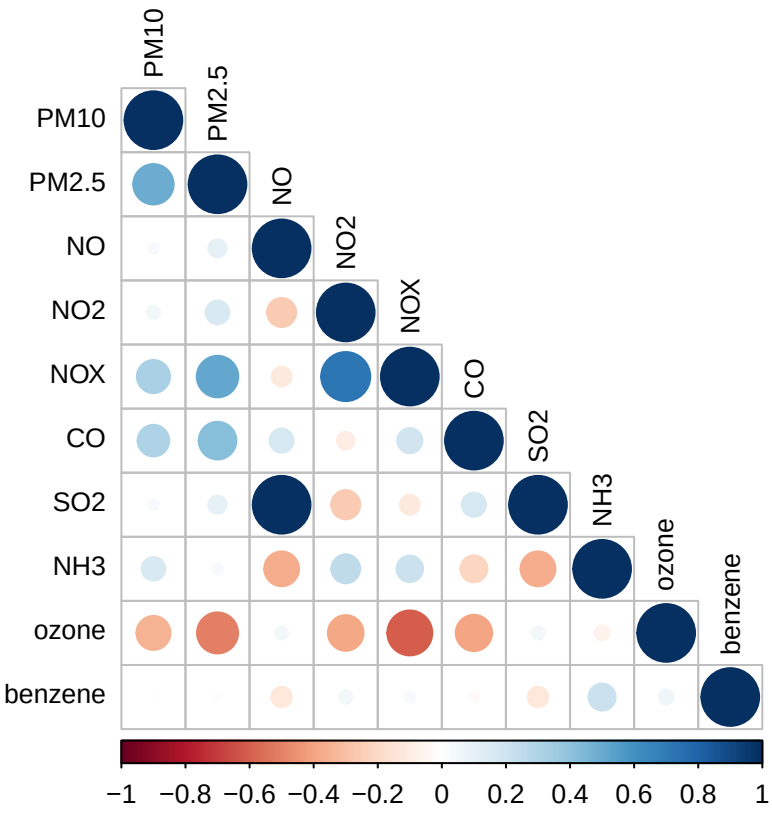
Key points :-

- The data shows that the concentration of PM10 fluctuates significantly around a certain value in different months. At regular intervals, there are spikes in PM10 concentration, possibly indicating blasting activities. Furthermore, there is a considerable amount of missing data in January and February. Towards the end of April, the data starts fluctuating around lower values, which may be attributed to specific measures taken to control air pollution during the summer.
- Similarly, PM2.5 particles exhibit a similar trend to PM10. One interesting observation is that PM2.5 has the minimum number of missing values, making it useful for univariate time series forecasting and potentially identifying blasting times in coal mining.

- In January, NH3 values initially show higher values but decrease towards the end of the month, remaining relatively constant afterward. This pattern suggests an intervention or action taken regarding NH3.
- SO2 particle values fluctuate around a constant mean value.
- Ozone particle values appear to be highly random and do not exhibit a consistent pattern. Therefore, it may not be worthwhile to use ozone values for further analysis.
- During the descriptive analysis, it was noted that benzene has only six unique values, with most of the data points fluctuating around 0.1 and 0.2. Additionally, data for February, March, and April is missing for benzene. This variable displays a different trend compared to other particles. Hence, it may be appropriate to exclude the ozone variable in multivariate analysis.

Correlation plot :- Let see correlation between variables.

```
library(zoo)
library(corrplot)
df <- dat
df[,c(4,5,6,7,8,9,10,11,12,13)] <- na.spline(df[,c(4,5,6,7,8,9,10,11,12,13)])
sdf <- df[,c(4,5,6,7,8,9,10,11,12,13)]
M<-cor(df[,c(4,5,6,7,8,9,10,11,12,13)])
corrplot(M, method="circle",type = "lower",col = NULL,
         title = "",tl.col = "black",
         tl.cex = 0.8)
```

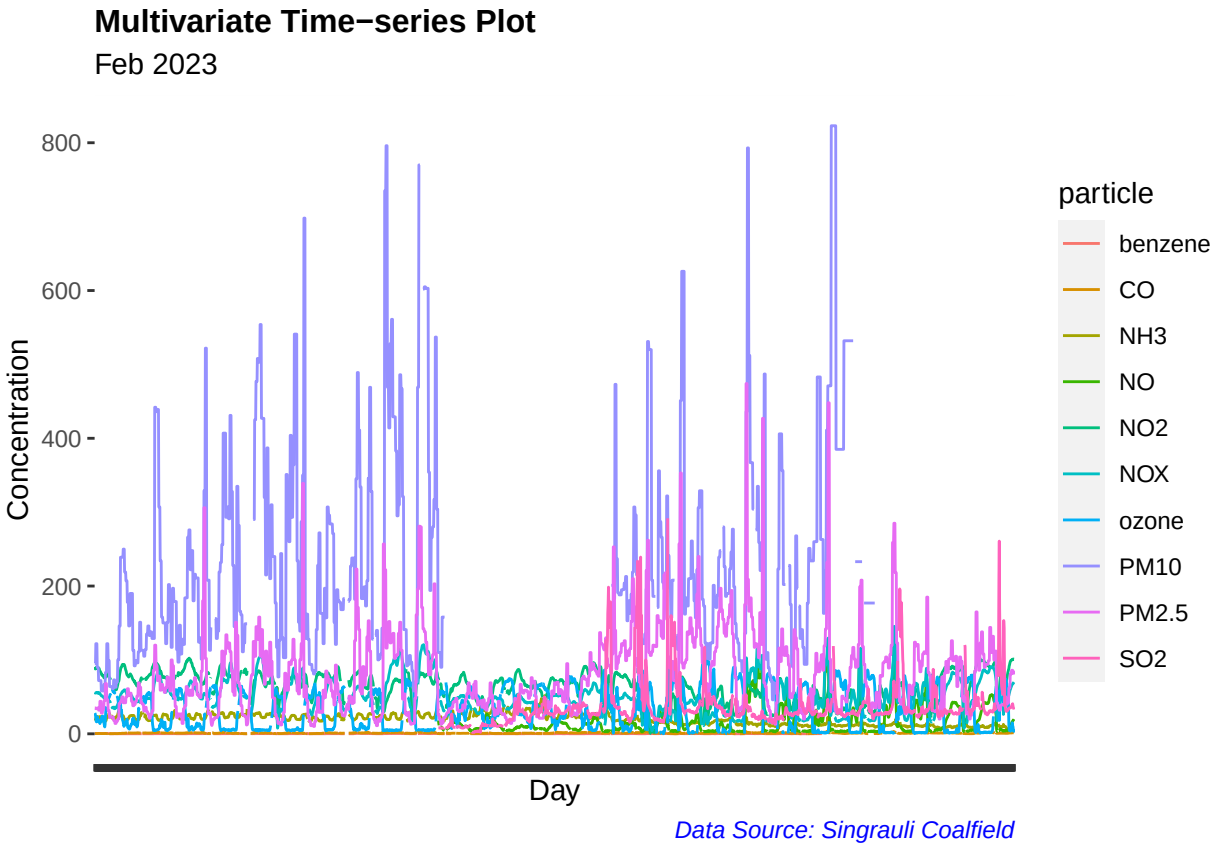


Key points :-

- There is a strong positive correlation between NO and SO2 particles.
- PM10 and PM2.5 particles, as well as PM2.5 and NOX particles, exhibit moderate correlation.
- Benzene shows extremely low correlation with all other particles, confirming the validity of our hypothesis to exclude it from further analysis.
- Ozone displays negative correlations with other variables.

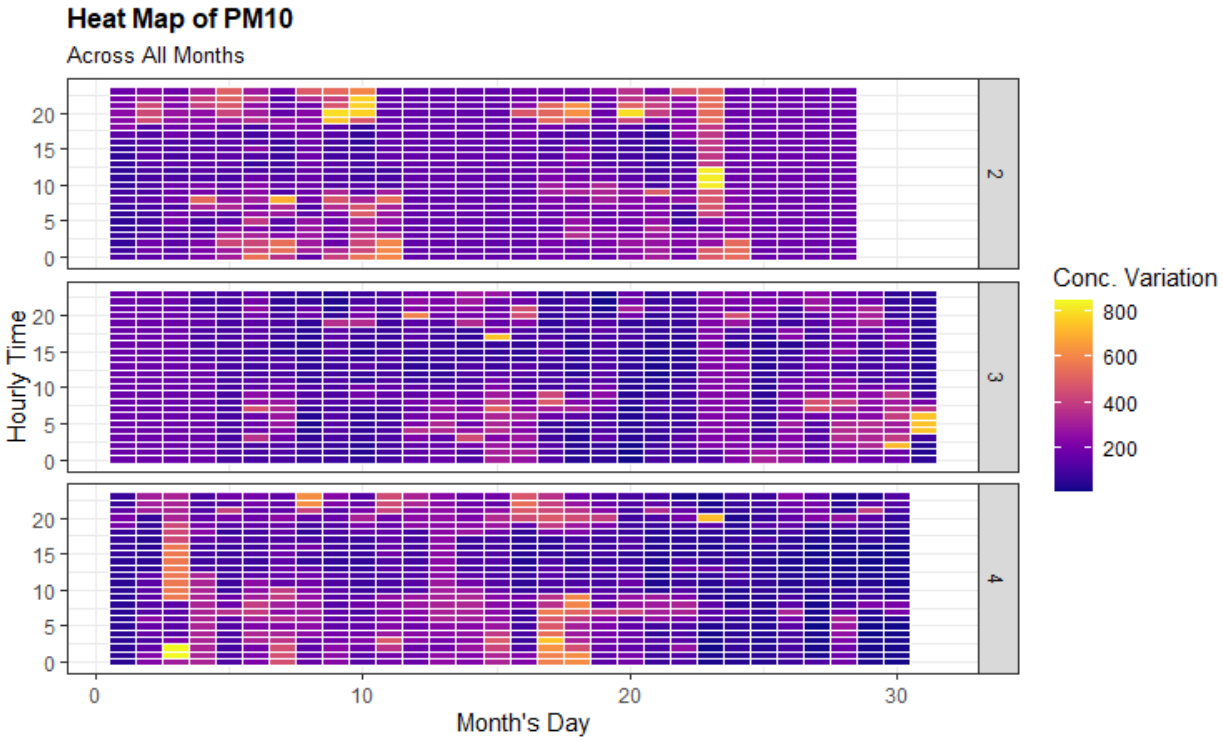
Multivariate Time-Series plot

```
library(ggplot2)
library(reshape2)
library(dplyr)
feb <- dat[which(dat$month==2),]
feb <- feb[ ,c(-14,-15,-16,-1,-3)]
# convert feb data in desired format as above
df <- dat
row.names(feb) <- feb$From
feb <- feb[ ,-1]
cfeb <- data.frame(particle = colnames(feb))
#cfeb <- cbind(cfeb, as.numeric(feb[1,]))
for(i in 1:2688){
 # y <- paste(as.character(feb[i,1]))
  cfeb <- cbind(cfeb, as.numeric(feb[i,]))
}

dum_feb <- dat[which(dat$month==2),]

colnames(cfeb)[2] <- "sawre"
colnames(cfeb)[2] <- as.character(df$From[1])
for(i in 2:2689){
  colnames(cfeb)[i] <- as.character(dum_feb$From[i-1])
}
cfeb_melt <- melt(cfeb)
ggplot(data=cfeb_melt, aes(x=variable, y=value, group = particle,
                                              colour = particle))+ geom_line() +labs(y= "Concentration", x
  theme(axis.text.x = element_blank())+
  labs(title = "Multivariate Time-series Plot",
       subtitle = "Feb 2023",
       caption = "Data Source: Singrauli Coalfield")+
  theme(
    plot.title = element_text(color = "Black", size = 12, face = "bold"),
    plot.subtitle = element_text(color = "black"),
    plot.caption = element_text(color = "blue", face = "italic")
  )
```



Key point :- The plot for February 2023 illustrates the contribution of different particles to air pollution. It is evident that PM10, PM2.5, and SO2 particles have a significant presence and contribute the most to air pollution. The concentrations of other particles are relatively lower compared to these three. Particularly, benzene shows a non-significant presence in the overall air pollution levels during that period.

Heat Map



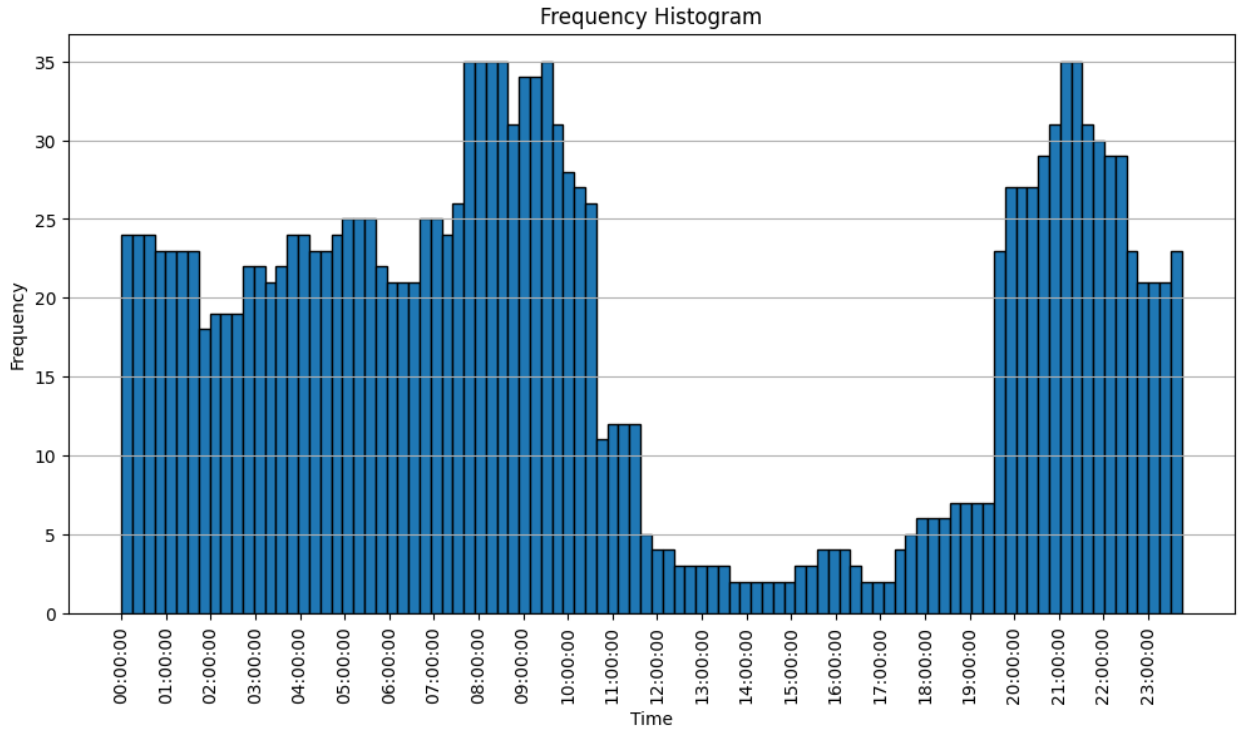
key point :-The plot for the months of February, March, and April in 2023 (represented by 2, 3, and 4 respectively) reveals distinct variations in air pollution levels. The presence of high concentration values across different days indicates that there is no fixed blasting time on a regular basis. This plot enables us to easily determine the concentration status on specific days and hours, providing valuable insights into the fluctuating nature of air pollution levels.

Detecting Blasting Time

To identify potential blasting times, I focused on the variable PM10. I extracted all the observed values and set a threshold based on the 3rd quantile. Next, I collected all the values that exceeded this threshold and recorded their corresponding time stamps. By doing so, I obtained a frequency table for each day, encompassing time intervals of 15 minutes and the frequency of high PM10 values observed during those intervals. Finally, I plotted the frequency distribution to visualize the pattern of occurrence.

```
load("C:/Users/psand/OneDrive/Desktop/EE798_FISA/Assignment_1/cleaned_data3.Rdata")

df <- dat

# First I load the cleaned_data.Rdata file
# where first time I will separete time component separately
load("C:/Users/psand/OneDrive/Desktop/EE798_FISA/Assignment_1/cleaned_data.Rdata")
df$Timing <- dat$tem[1:8640]
load("C:/Users/psand/OneDrive/Desktop/EE798_FISA/Assignment_1/cleaned_data3.Rdata")

#  Approach for getting blast time
index <- which(df$PM10>238) # greater than 3rd quantile
#length(index) # total of 1721 entries
```

```
#unique(df$Timing[index])
#table(df$Timing[index])
library(plyr)
y <- count(df$Timing[index])
#y$freq
#mean(y$freq)
#summary(y$freq)
ind <- which(y$freq>25)
library(dplyr)
y <- y %>% arrange(freq)
y <- y %>% arrange(x)

# y contains total 96 entries it means it covers
# whole day
```



Based on the frequency distribution, it appears that blasting times are more dispersed during early morning and night hours. It is likely that blasting events occur more frequently between 7:45:00 AM to 9:45:00 AM and 19:45:00 PM to 21:45:00 PM. Afternoon times show a lower probability of blasting occurrences. However, it should be noted that the government guidelines for blasting time are from 13:45:00 to 14:45:00, which is rarely followed in Singrauli Coalfield.

Assignment Question:

Find the probability of an open-pit blast happening between 14:15 to 14:30?.

Solution: Since the distribution type in the given plot is unclear, we will approach this problem using the frequency approach to probability.

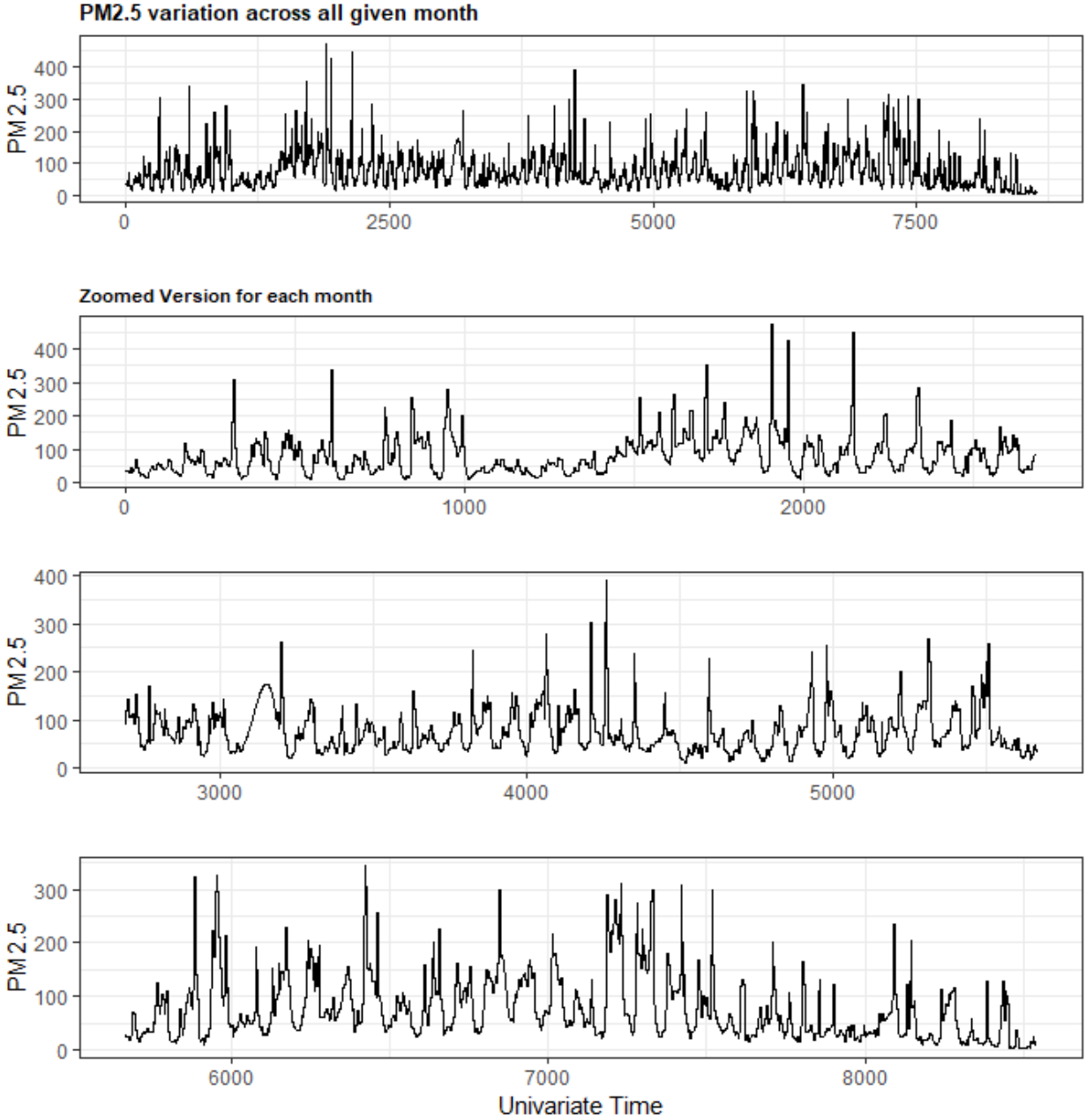
$$P(Blasting\ occured\ between\ 14:15\ to\ 14:30) = 2/1721 = 0.001$$

Note **-** For previous plot of frequency I used `Python`. Below is code snapshot.

```
import pyreadr
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

result = pyreadr.read_r('/content/blast_time.Rdata')
df = result["y"]

x = df[["x"]]
freq = df[["freq"]]
plt.figure(figsize=(12, 6))
plt.hist(x, bins=96, weights=freq, edgecolor='black')
# Formatting the plot
plt.title("Frequency Histogram")
plt.xlabel("Time")
plt.ylabel("Frequency")

hour_labels = ['00:00:00', '01:00:00', '02:00:00', '03:00:00', '04:00:00', '05:00:00',
               '06:00:00', '07:00:00', '08:00:00', '09:00:00', '10:00:00', '11:00:00',
               '12:00:00', '13:00:00', '14:00:00', '15:00:00', '16:00:00', '17:00:00',
               '18:00:00', '19:00:00', '20:00:00', '21:00:00', '22:00:00', '23:00:00']
plt.xticks( hour_labels, rotation=90)

plt.grid(axis='y')
```

Time- Series Modeling - Forecasting

For forecasting purposes, I selected the univariate time series of PM2.5 pollutant. This variable has the minimum number of missing values, and after applying cubic spline interpolation to fill in the missing values, the overall trend of the PM2.5 time series plot remains largely unchanged. This approach enables the creation of a more accurate model using the original data. Additionally, PM2.5 is positively correlated with PM10, NOX, and CO, suggesting that this model can be valuable in estimating missing values for these variables as well.



The time-series plot of PM2.5 exhibits repetitive cycles at regular intervals. To forecast its future values, I have opted to use the SARIMA (Seasonal Auto Regressive Integrated Moving Average) model.

Libraries used - `Forecast` & `tseries`

```
library(forecast)
library(tseries)

# applying model on first 8000 entries
pm2.5 <- dat[ ,c(2,5)]
pm2.5 <- pm2.5[c(1:8000),]

# Converting it in time series data format
spm2.5 <- ts(data=pm2.5$PM2.5,start =1,end = 8000,frequency = 1 )
class(spm2.5)
```

```
## [1] "ts"
```

Checking for stationarity by Augmented Dickey-Fuller Test.

```
adf.test(spm2.5)
```

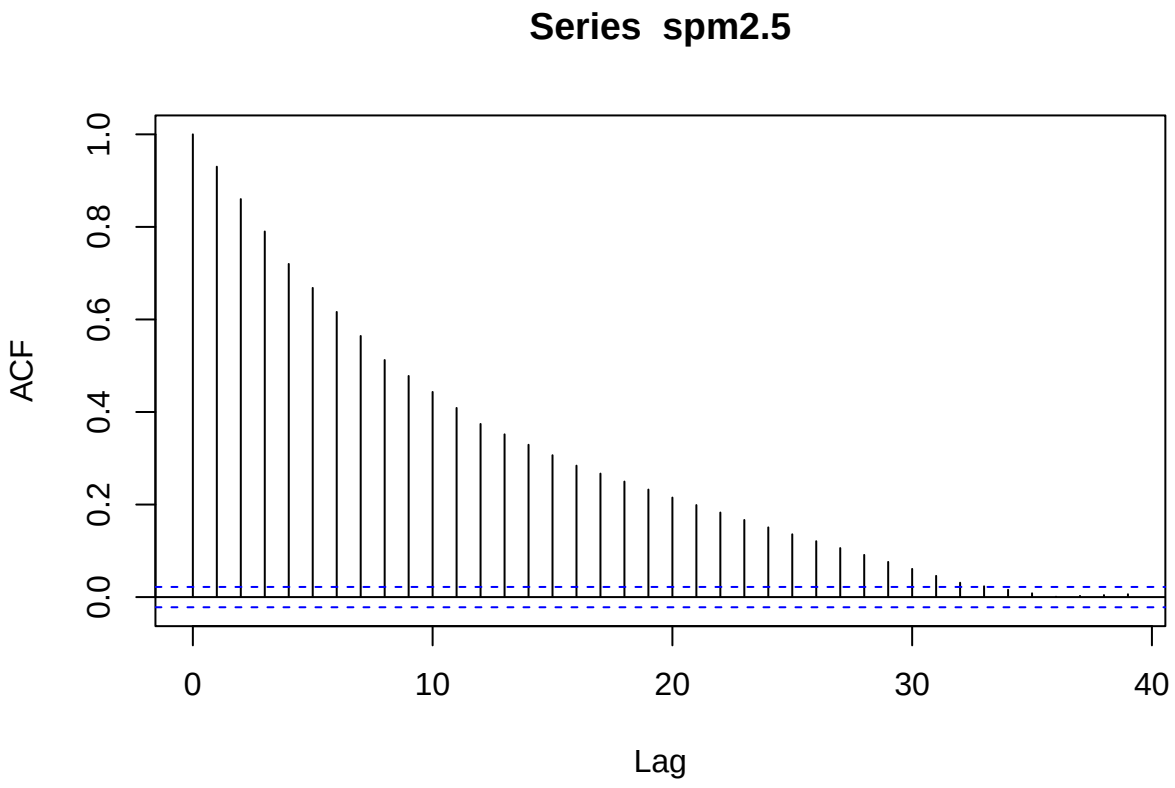
```
## Warning in adf.test(spm2.5): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  spm2.5
## Dickey-Fuller = -13.647, Lag order = 19, p-value = 0.01
## alternative hypothesis: stationary
```

Since p-value is less than 0.05 we fail to accept alternative hypothesis of stationarity.

Auto-Correlation Function (ACF) plot

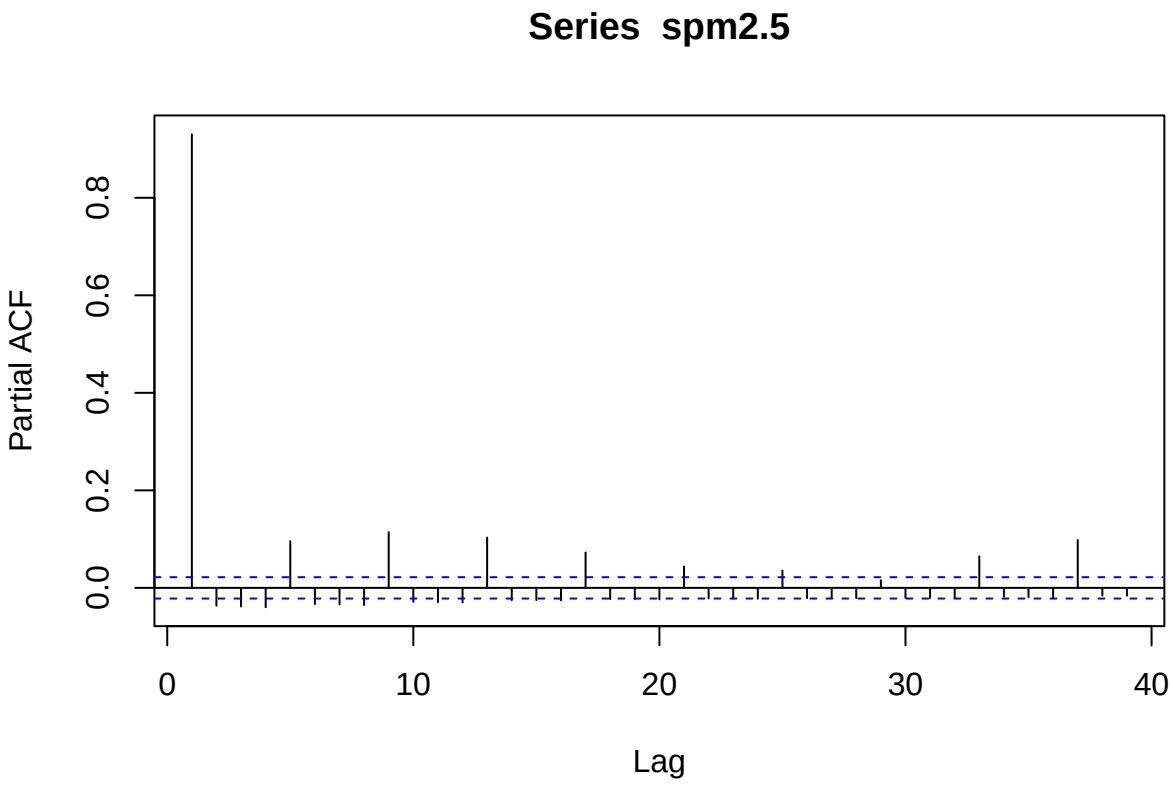
```
acf(spm2.5)
```



In this plot we can see there is trend in correlation till lag 30. This show that series is stationarity and from there we can also get order of SARIMA.

Partial-autocorrelation plot (PACF) plot

```
pacf(spm2.5)
```



In this plot, at every 4th lag spike is crossing blue line this show some seasonality. And from that I get order of SARIMA.

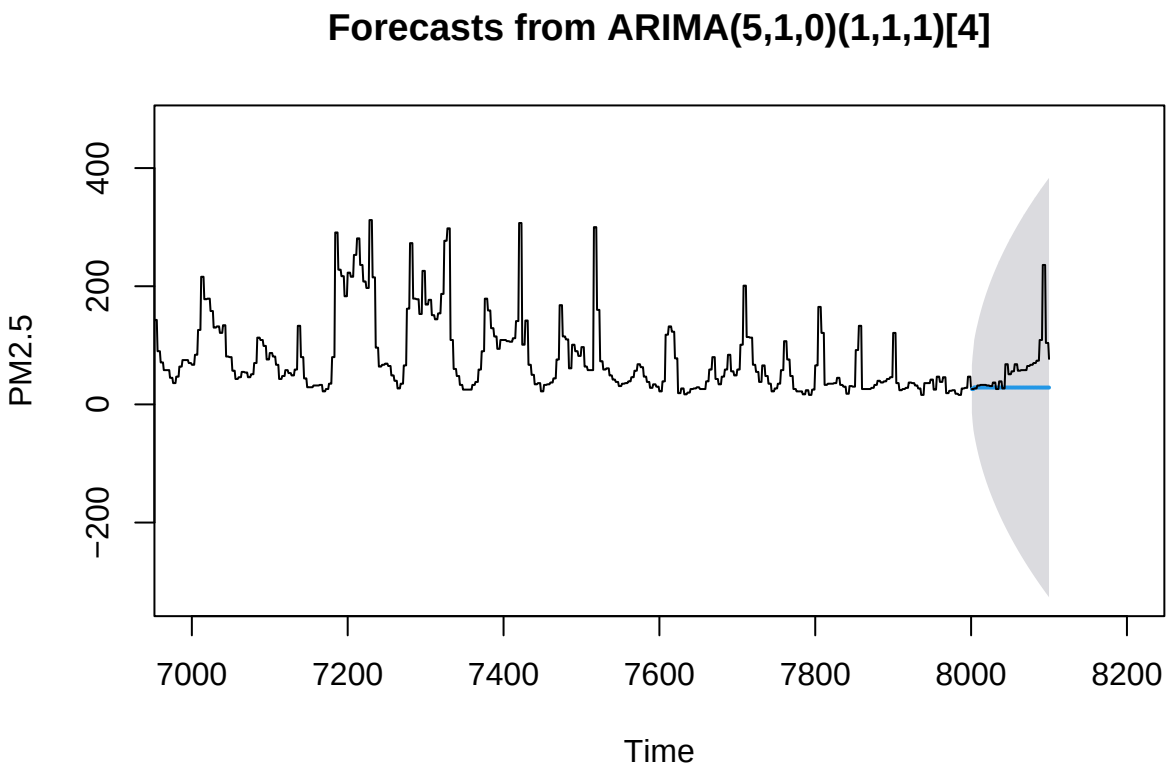
Since our series non-stationary so to make it stationary I take first difference. It implies $d = 1$ in model.

For finding Seasonal order I used below code and get order c(1,1,1).

```
# Determine the seasonal differences of the time series data
seasonal_diffs <- diff(spm2.5, differences = 1, lag = 4)  # Replace 'D' and 's' with appropriate values
#plot(seasonal_diffs)
library(ggplot2)
# Plot the seasonal autocorrelation function (SACF)
#acf(seasonal_diffs, lag.max = 40)
# Plot the seasonal partial autocorrelation function (SPACF)
#pacf(seasonal_diffs, lag.max = 40)
```

Model code

```
sarima_model <- arima(spm2.5,order = c(5, 1, 0),seasonal = list(order = c(1, 1, 1), period = 4))

sarima_model_f=forecast(sarima_model,level = c(95),h=100)
plot(sarima_model_f,xlim=c(7000,8200),xlab = "Time",ylab = "PM2.5")
lines(x=c(8001:8100),y=dat$PM2.5[8001:8100])
```



In the above plot I forecast next 100 value of PM2.5 i.e. predict the PM2.5[8001:8100]

Validation using BOX-test

```
Box.test(sarima_model_f$residuals, lag=8, type= "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  sarima_model_f$residuals
## X-squared = 198.95, df = 8, p-value < 2.2e-16
```

```
Box.test(sarima_model_f$residuals, lag=7, type= "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  sarima_model_f$residuals
## X-squared = 0.99222, df = 7, p-value = 0.995
```

In first test gives p-value nearly 0. Residuals are independent after lag 8 before that they are not independent. OR we can see our predicted model show autocorrelation before lag 8 not after that.

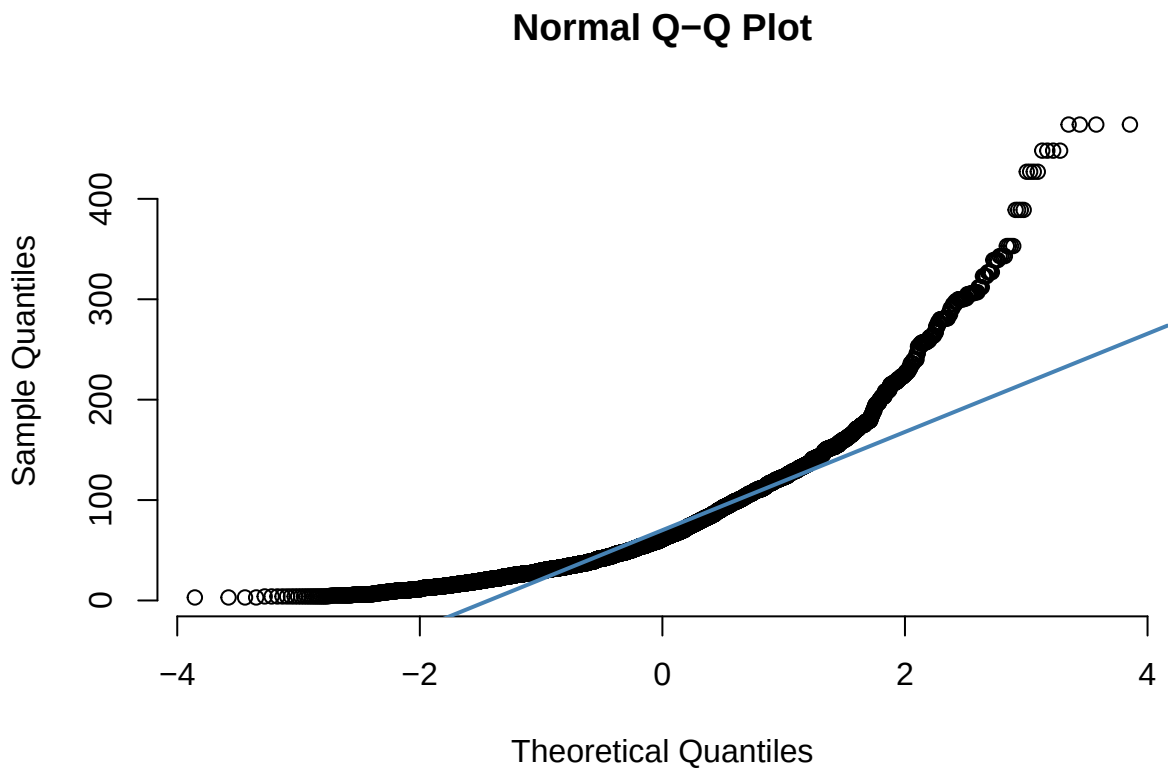
Distribution Modeling

Lets take PM2.5 particle. I take first 5000 data points as training data.

```
y <- dat$PM2.5 #8640 entries
y <- y[-c(50001:8650)] # First 5000
```

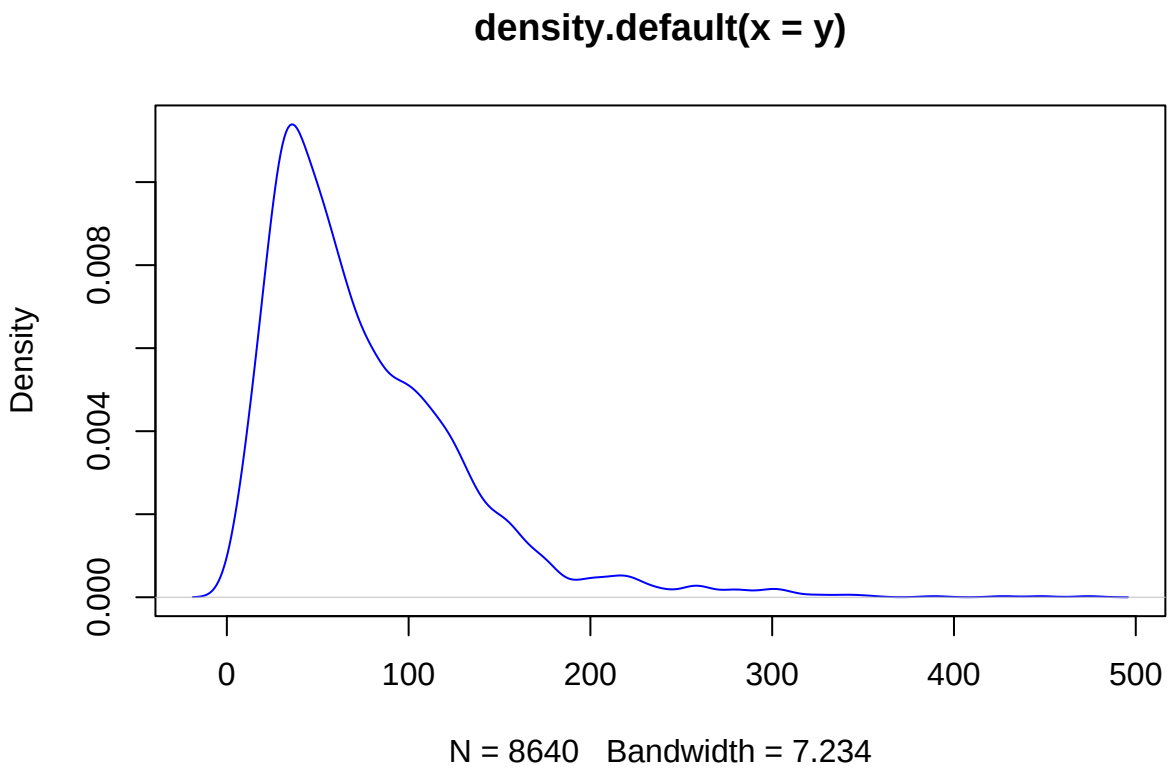
Quantile- Quantile plot for vector y.

```
qqnorm(y, pch = 1, frame = FALSE)
qqline(y, col = "steelblue", lwd = 2)
```



From the graph we infer that data points are positively skewed. And we can also see it by `density()` function of R.

```
plot(density(y),col="blue")
```

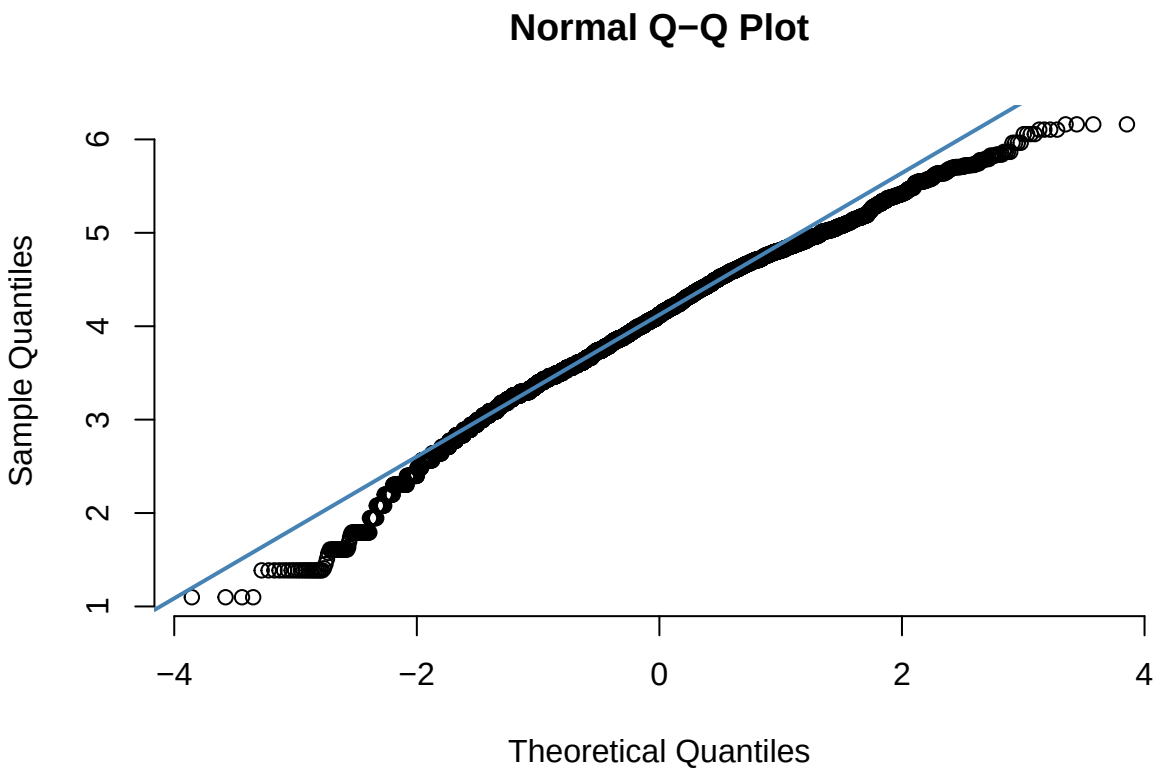


Taking logarithm of vector y.

```
log.y <- log(y)
```

Quantile-Quantile plot for log.y .

```
# quantile quantile plot
qqnorm(log.y, pch = 1, frame = FALSE)
qqline(log.y, col = "steelblue", lwd = 2)
```



In this graph most of points lies on blue straight line. This implies roughly `log.y` follow normal distribution.

Estimating the parameter :-

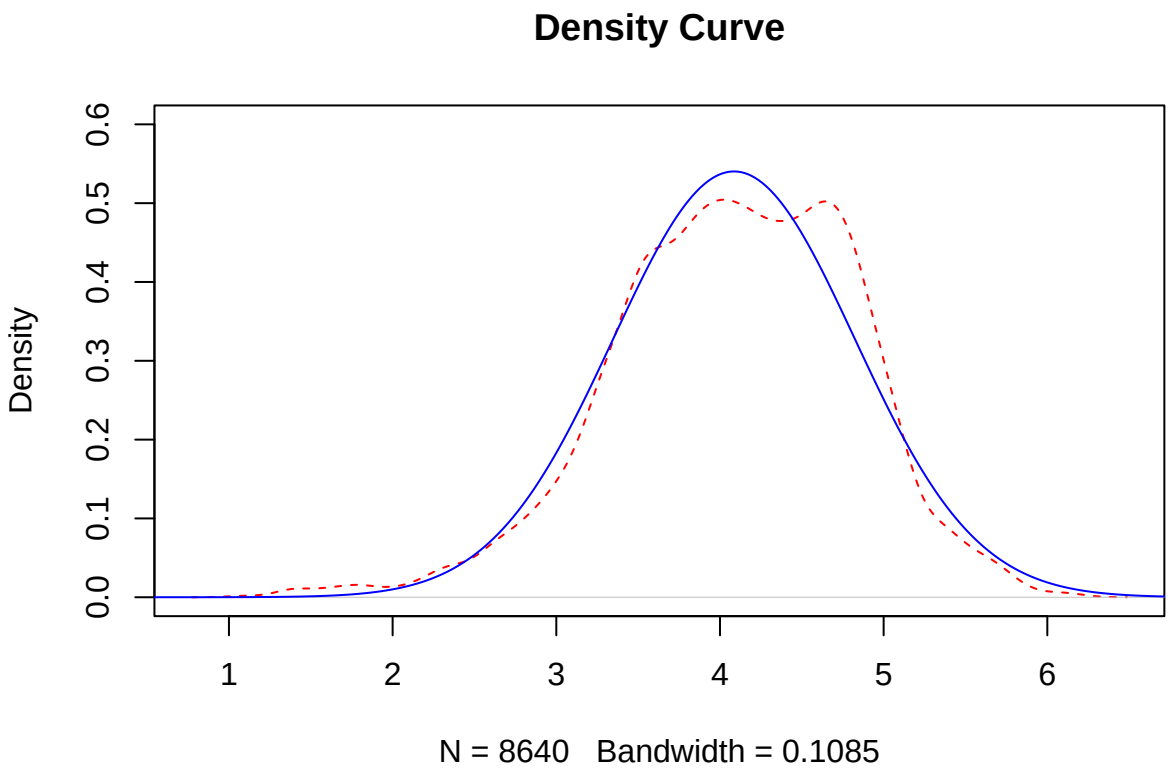
Let $Y_1, Y_2, Y_3, ...Y_{5000}$ are the points of `log.y` vector. We know these points follow $N(\mu, \sigma^2)$. Assume these points are IID. We have estimate μ and σ^2.

For estimating these parameter, We used *Maximum-Likelihood estimator*.

By MLE μ equals to the mean of sample and σ^2 equal to the variance of sample.

Keeping this in mind, let's plot the density curve of a normal distribution alongside our sample.

```
x <- seq(-10,10,length=1000)
y_x <- dnorm(x,mean = mean(log.y),sd=sd(log.y))
plot(density(log.y),ylim=c(0.0,0.6),col="red",main = "Density Curve",type = "l",lty=2)
lines(x=x,y=y_x,col="blue")
```



Blue line for estimated density and red line for actual points. This graph depict that `log.y` follow Normal distribution with mean

```
mean(log.y)
```

```
## [1] 4.085848
```

and standard deviation

```
sd(log.y)
```

```
## [1] 0.7384103
```

Result :- Since `log.y` follow normal distribution then `y` follow log normal distribution.
