TSA

Yuxiang Ren

#packges

```
library(readxl)
#library(xlsx) #this package doesn't work on Macs very well
library(tidyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(ggplot2)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(Kendall)
library(tseries)
library(outliers)

library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr   1.1.2     v readr   2.1.4
## v forcats 1.0.0     v stringr 1.5.0
## v purrr   1.0.1     v tibble  3.2.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(smooth)
```

```
## Loading required package: greybox
## Package "greybox", v1.0.8 loaded.
##
##
## Attaching package: 'greybox'
##
## The following object is masked from 'package:lubridate':
##
##     hm
##
## The following object is masked from 'package:tidyr':
##
##     spread
##
## This is package "smooth", v3.2.0
```

```
library(zoo)
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
##
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
library(dplyr)
library(smooth)
#install.packages("grid")
library(grid)
```

#Notice I don't know where the conflict occurred, the checkresiduals run of chunk4 can't come out. But the whole document can be knit

#data

```
raw <- read.csv("./Data/Processed/PM2.5_daily_city_2000_2021.csv")
Table_raw <- raw[c(1:20),]
kable(Table_raw,caption = "Raw data")
```

```
raw_wider <- raw %>% pivot_wider(names_from = city_id, values_from = meanpm) %>%
  mutate(date = as.Date(paste(year, month, day, sep = "-"))) %>%
  select(date, everything())
```

Table 1: Raw data

year	month	day	city_id	meanpm
2000	1	1	1100	130.86416
2000	1	1	1200	148.09160
2000	1	1	1301	144.00916
2000	1	1	1302	148.45776
2000	1	1	1303	130.70928
2000	1	1	1304	143.41543
2000	1	1	1305	144.08262
2000	1	1	1306	118.38928
2000	1	1	1307	66.89358
2000	1	1	1308	92.74160
2000	1	1	1309	134.39611
2000	1	1	1310	147.80079
2000	1	1	1311	142.80690
2000	1	1	1401	64.93789
2000	1	1	1402	54.55570
2000	1	1	1403	75.32231
2000	1	1	1404	90.57612
2000	1	1	1405	95.47729
2000	1	1	1406	48.40086
2000	1	1	1407	78.33466

#Beijing 1100 ##data

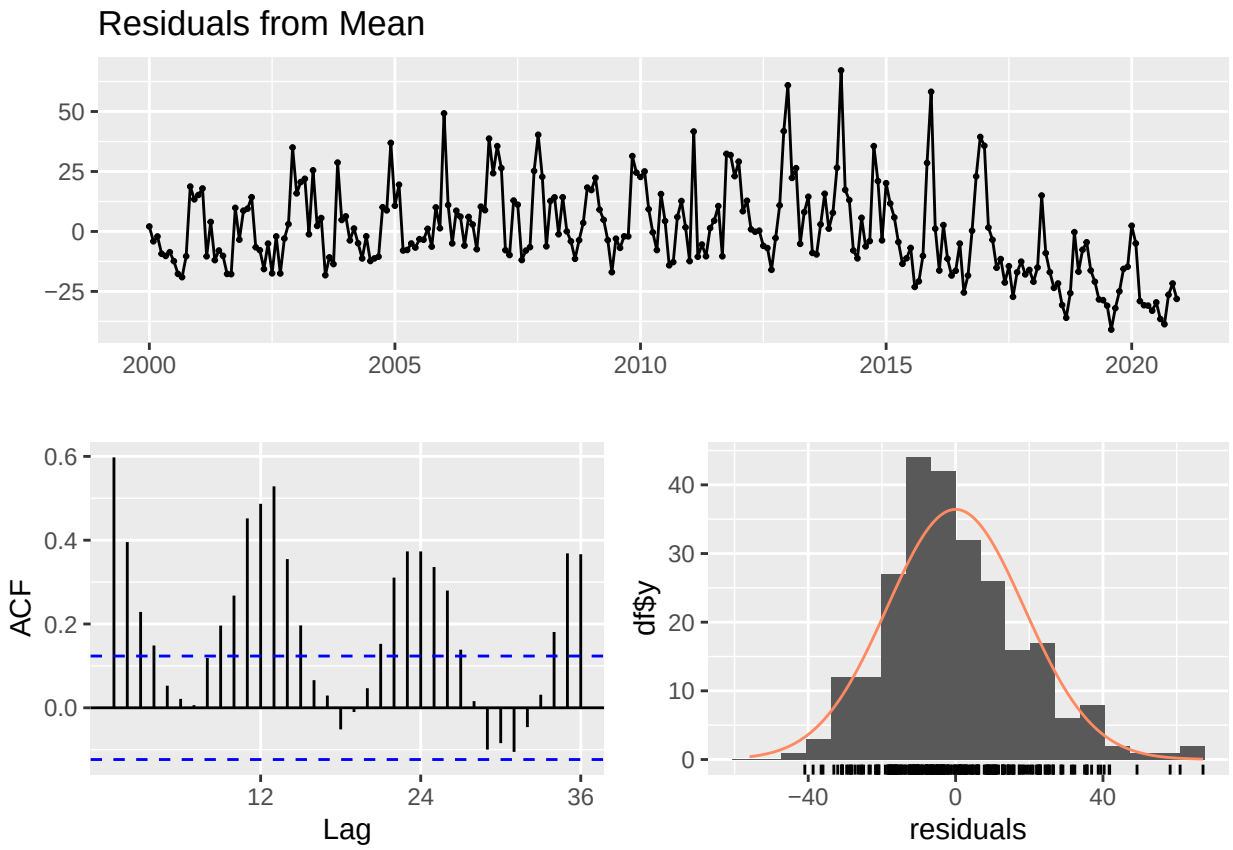
```
Beijing <- raw_wider[,c('date', 'year', 'month', 'day', '1100')] %>% rename(pm2.5 = '1100') %>% group_by
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#question, mean PM or accumulated PM

Beijing_train <- Beijing %>% filter(date < as.Date("2021-01-01"))
Beijing_test <- Beijing %>% filter(date >= as.Date("2021-01-01"))

ts_Beijing_train <- ts(Beijing_train$monthlyPM,
                       start=c(2000,1), frequency = 12)
ts_Beijing_all <- ts(Beijing$monthlyPM,
                       start=c(2000,1), frequency = 12)
```

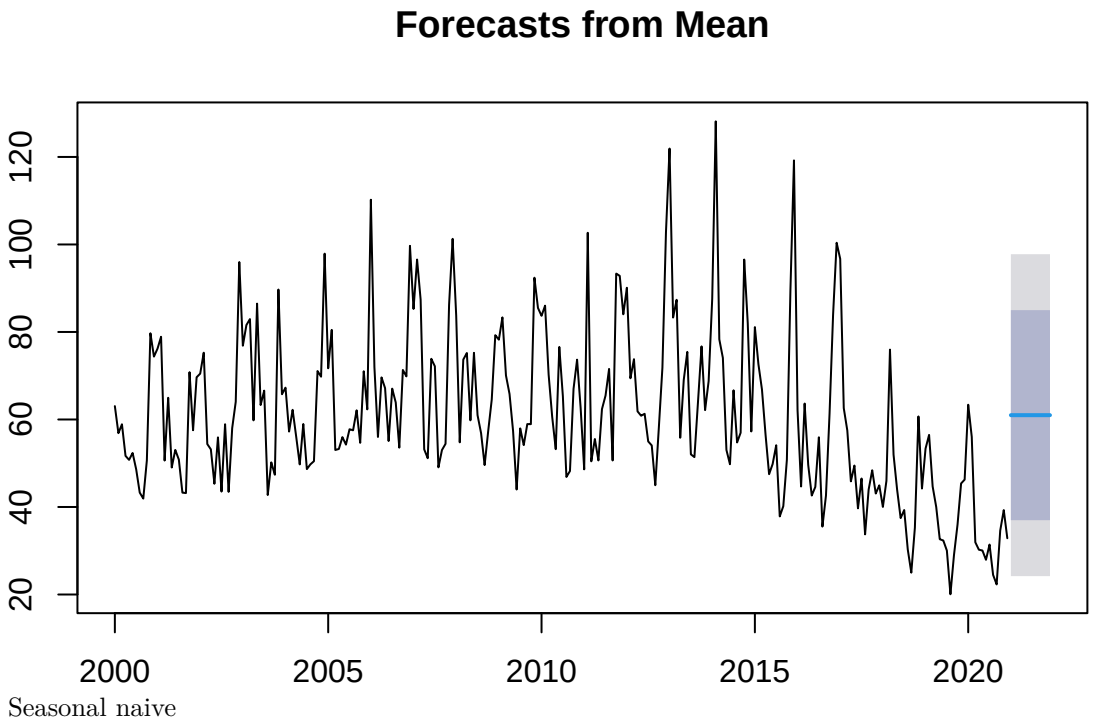
##Model 1: Arithmetic mean

```
AM_Beijing <- meanf(y = ts_Beijing_train, h = 12)
checkresiduals(AM_Beijing)
```



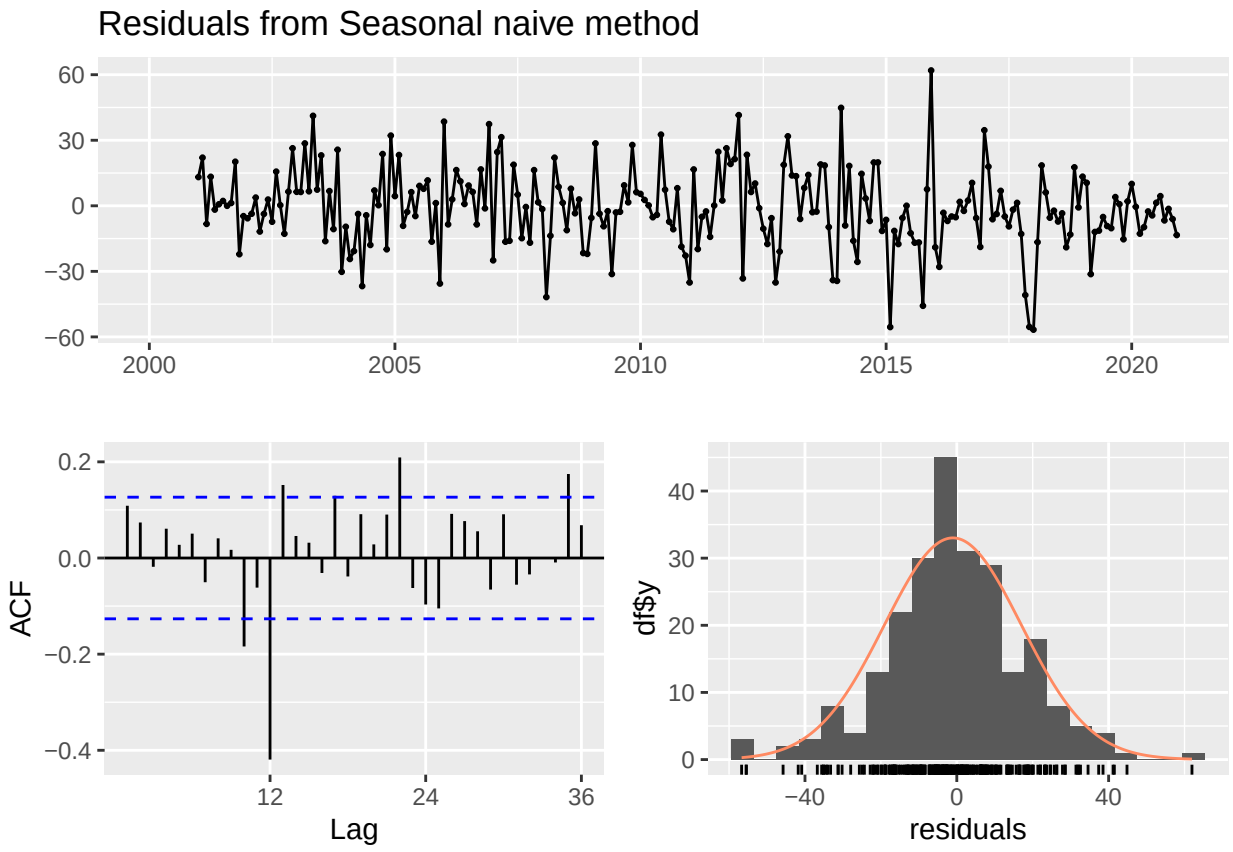

```
##
##  Ljung-Box test
##
## data:  Residuals from Mean
## Q* = 534.57, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
plot(AM_Beijing)
```



Model 2: Seasonal naive

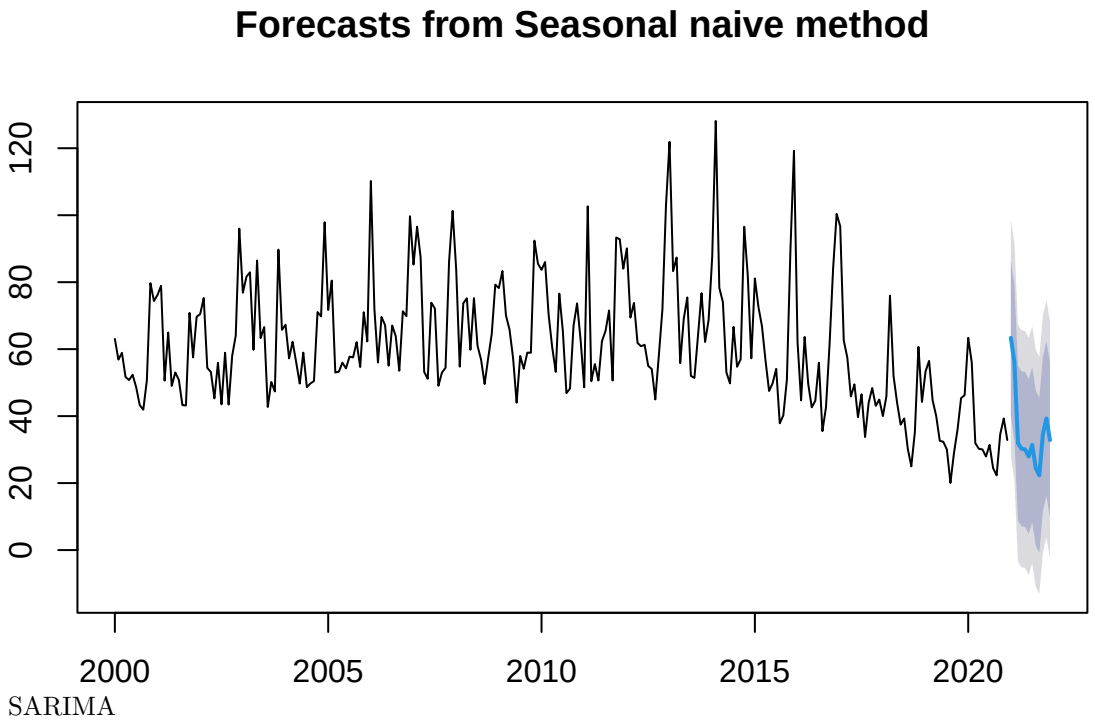
```
SNAIVE_Beijing <- snaive(ts_Beijing_train, h=12, holdout=FALSE)
checkresiduals(SNAIVE_Beijing)
```



##

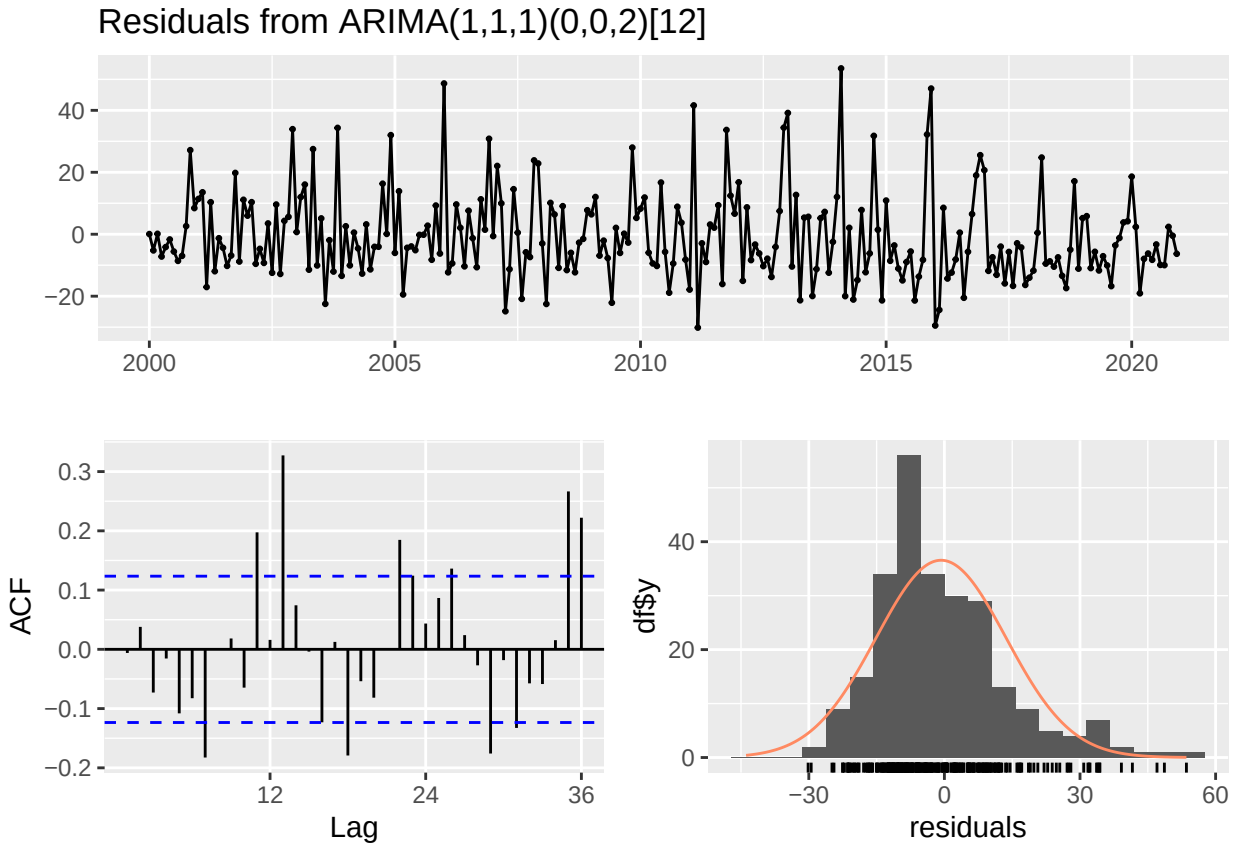
```
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 92.915, df = 24, p-value = 4.72e-10
##
## Model df: 0.   Total lags used: 24
```

```
plot(SNAIVE_Beijing)
```



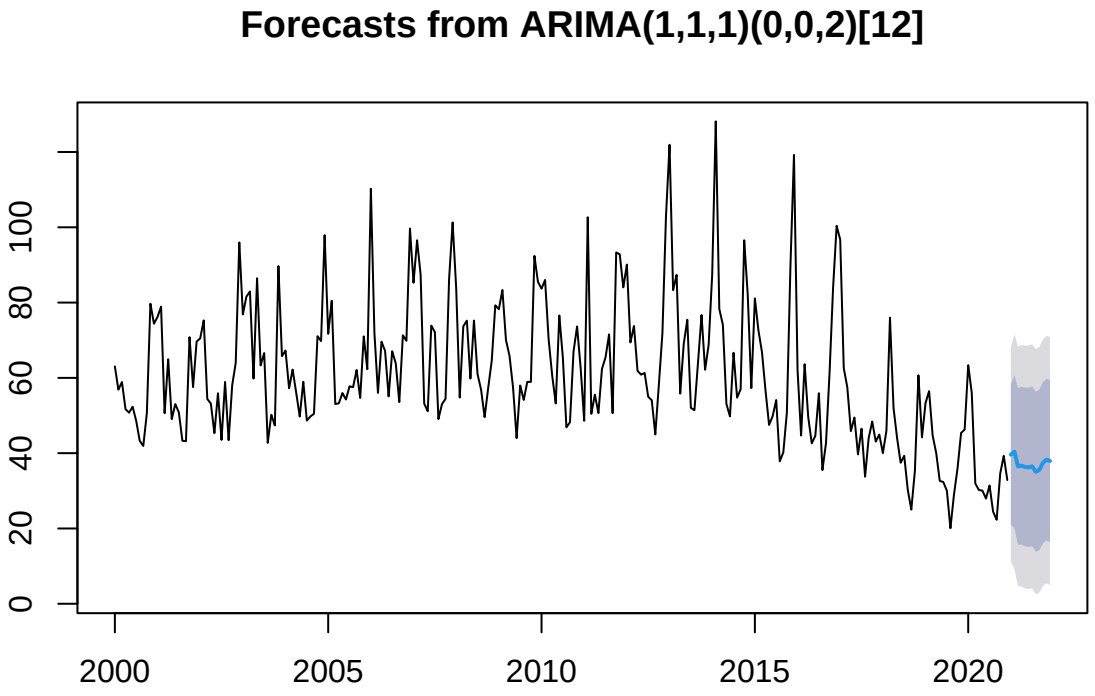
Model 3: SARIMA

```
SARIMA_Beijing_fit <- auto.arima(ts_Beijing_train)
checkresiduals(SARIMA_Beijing_fit)
```



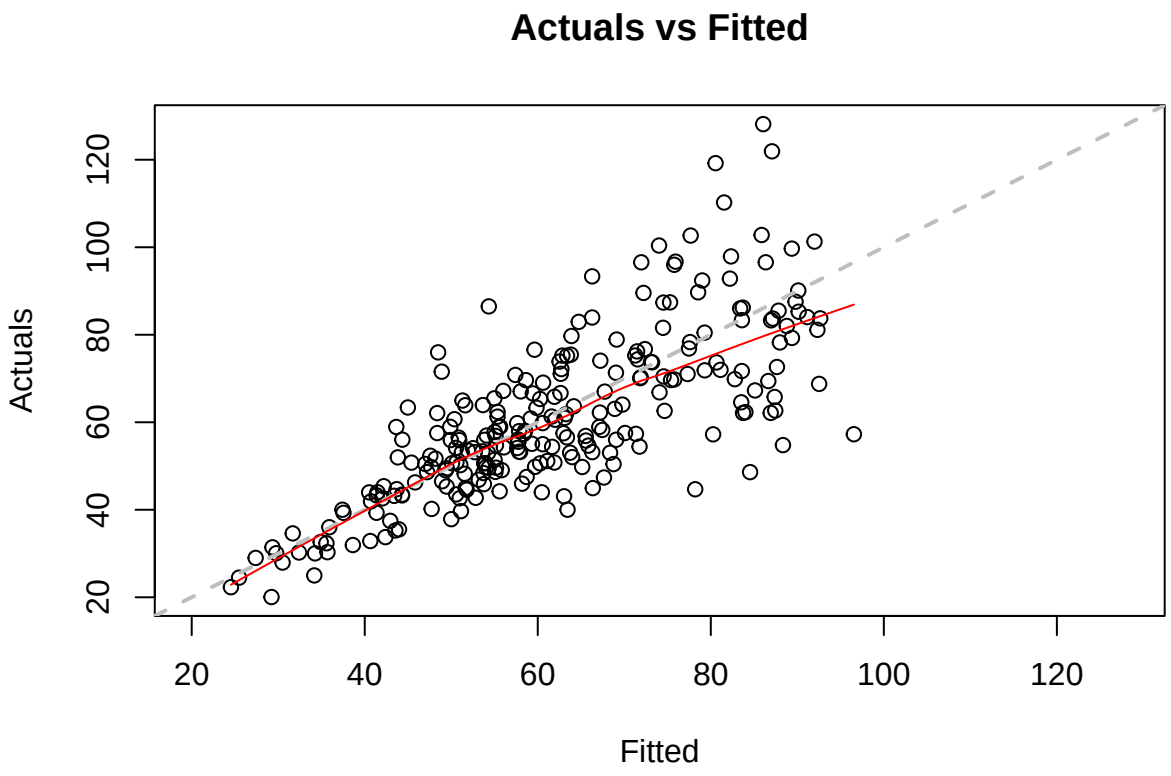

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)(0,0,2)[12]
## Q* = 87.011, df = 20, p-value = 2.458e-10
##
## Model df: 4.   Total lags used: 24
```

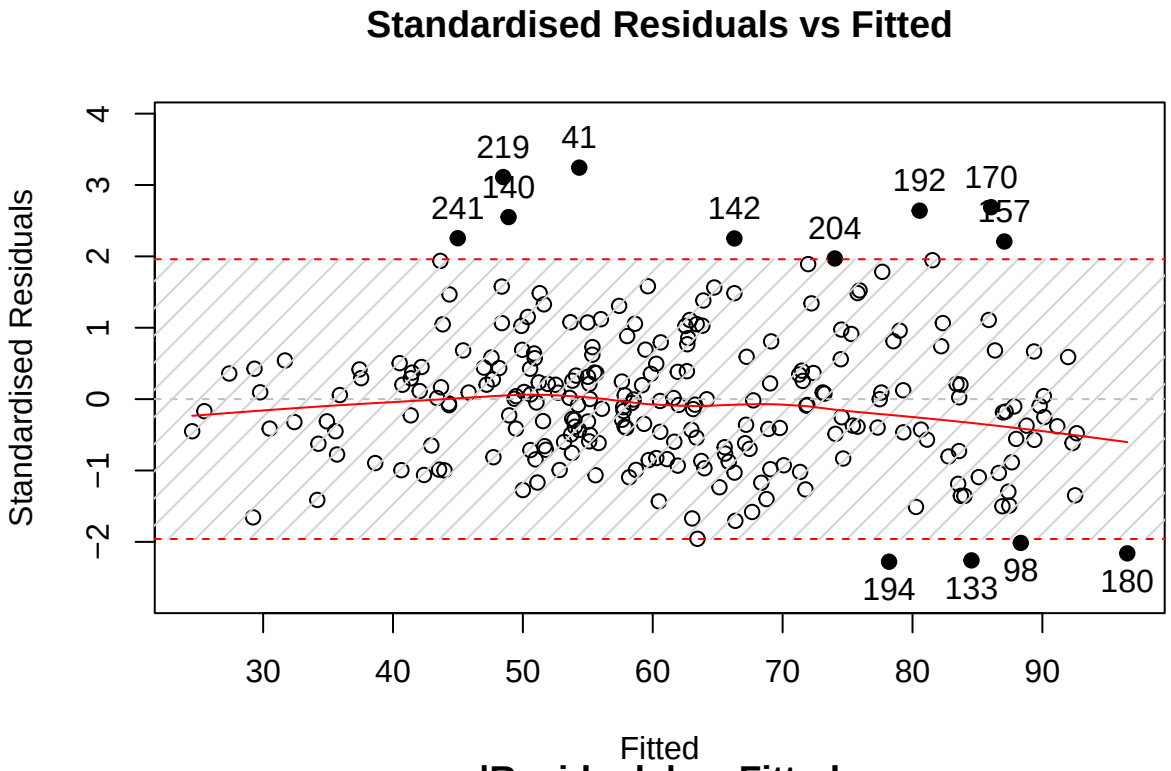
```
#Generating forecasts
#remember auto.arima does not call the forecast() internally so we need one more step
SARIMA_Beijing <- forecast(SARIMA_Beijing_fit,h=12)
plot(SARIMA_Beijing)
```

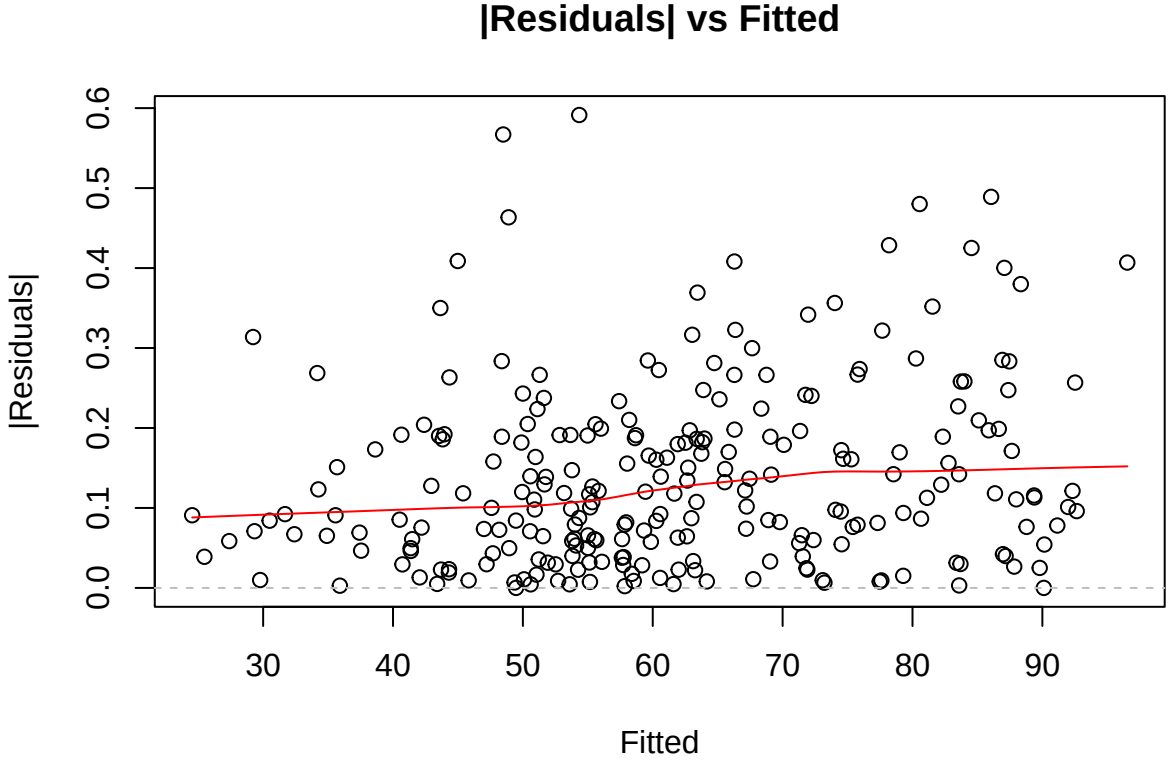


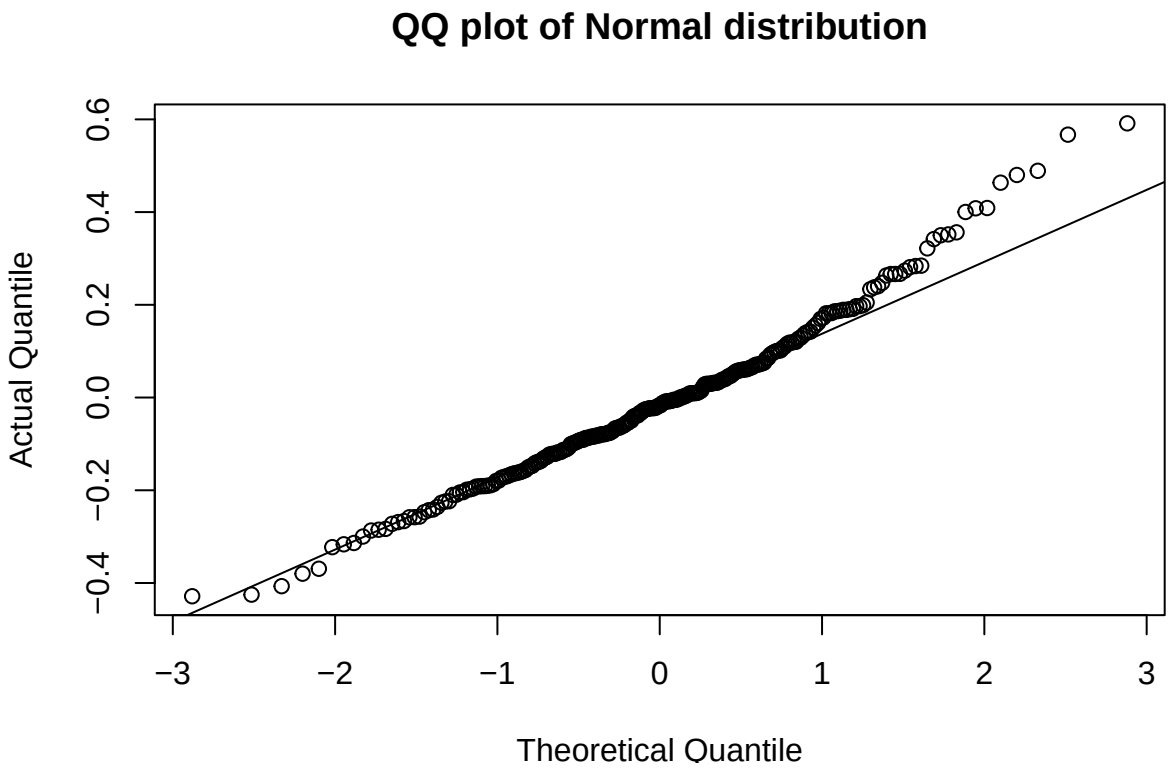
Model 4: seasonal exponential smoothing,Triple Exponential Smoothing

```
SSES_Beijing <- es(ts_Beijing_train,model="ZZZ",h=12,holdout=FALSE)
plot(SSES_Beijing)
```

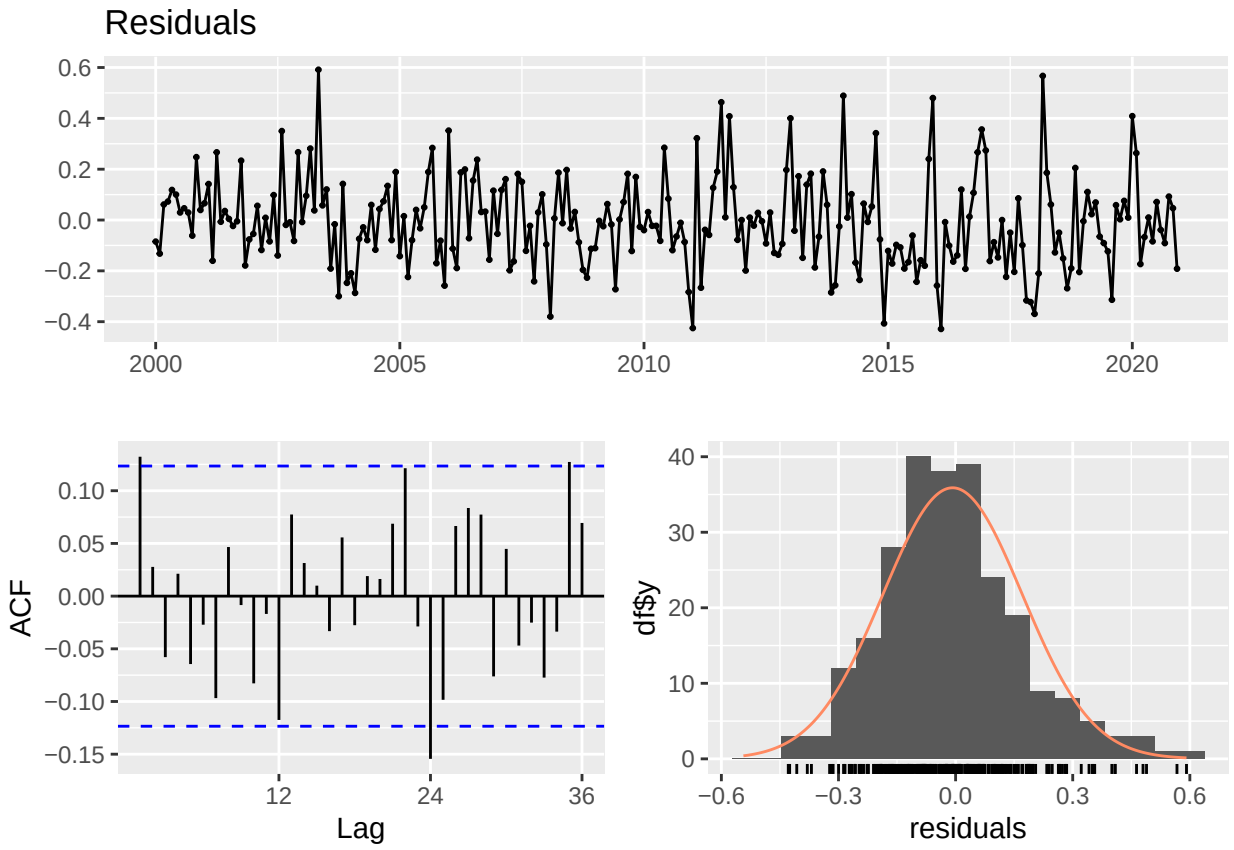



Standardised Residuals vs Fitted
Standardised Residuals
Fitted
219
41
140
241
142
204
192
170
157
194
133
98
180


|Residuals| vs Fitted
|Residuals|
Fitted




```
checkresiduals(SSES_Beijing)
```



```
##
##  Ljung-Box test
##
## data:  Residuals
## Q* = 31.261, df = 24, p-value = 0.1464
##
## Model df: 0.   Total lags used: 24
```

##Model 2+3

```
SNAIVE_Beijing$mean
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2021 63.37867 56.02063 31.93727 30.22966 30.06280 27.93552 31.41580 24.48430
##           Sep      Oct      Nov      Dec
## 2021 22.30550 34.60363 39.29200 32.85806
```

```
SARIMA_Beijing$mean
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2021 39.62250 40.40345 36.48258 36.66916 36.32987 36.25617 36.49963 35.06763
##           Sep      Oct      Nov      Dec
## 2021 35.54709 37.44673 38.29073 37.92379
```

```
df23 <- data.frame(SNAIVE_Beijing$mean, SARIMA_Beijing$mean)
colnames(df23) <- c("SNAIVE","SARIMA")

df23_mean <- df23 %>% mutate(SNAIVE_SARIMA = (SNAIVE + SARIMA)/2)
```

##Model 2+4

```
SSES_Beijing$forecast
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2021 39.07450 36.93737 31.24317 26.56132 24.47794 24.96635 24.13003 20.73591
##           Sep      Oct      Nov      Dec
## 2021 19.97270 25.93105 30.33946 32.56003
```

```
SARIMA_Beijing$mean
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2021 39.62250 40.40345 36.48258 36.66916 36.32987 36.25617 36.49963 35.06763
##           Sep      Oct      Nov      Dec
## 2021 35.54709 37.44673 38.29073 37.92379
```

```
df24 <- data.frame(SSES_Beijing$forecast, SARIMA_Beijing$mean)
colnames(df24) <- c("SSES","SARIMA")

df24_mean <- df24 %>% mutate(SSES_SARIMA = (SSES + SARIMA)/2)
```

##accuacy

Table 2: Forecast Accuracy for Seasonal Data

	ME	RMSE	MAE	MPE	MAPE
MEAN	-28.1059	31.6898	29.2560	-119.0707	120.7650
SNAIVE	-2.4987	13.4377	9.3796	-17.3618	28.6924
SARIMA	-4.3333	14.5863	12.2154	-32.3573	45.6127
SSES	4.8008	12.3008	7.7936	5.2446	19.6416

Table 3: Forecast Accuracy for Seasonal Data

	ME	RMSE	MAE	MPE	MAPE
MEAN	-28.105858	31.68980	29.256003	-119.070730	120.76499
SNAIVE	-2.498701	13.43770	9.379570	-17.361823	28.69245
SARIMA	-4.333324	14.58628	12.215420	-32.357291	45.61268
SSES	4.800801	12.30076	7.793640	5.244604	19.64161
SNAIVE_SARIMA	-3.416013	13.00090	9.967983	-24.859557	35.26887
SSES_SARIMA	0.233739	12.50184	9.574127	-13.556343	31.19364

```
MEAN_scores <- accuracy(AM_Beijing$mean,Beijing_test$monthlyPM)
SNAIVE_scores <- accuracy(SNAIVE_Beijing$mean,Beijing_test$monthlyPM)
SARIMA_scores <- accuracy(SARIMA_Beijing$mean,Beijing_test$monthlyPM)
SSES_scores <- accuracy(SSES_Beijing$forecast,Beijing_test$monthlyPM)
M2_3_scores <- accuracy(df23_mean$SNAIVE_SARIMA,Beijing_test$monthlyPM)
M2_4_scores <- accuracy(df24_mean$SSES_SARIMA,Beijing_test$monthlyPM)
# 4 models Table
Beijing_scores0 <- as.data.frame(rbind(MEAN_scores, SNAIVE_scores, SARIMA_scores,SSES_scores))

row.names(Beijing_scores0) <- c("MEAN", "SNAIVE","SARIMA","SSES")

kbl(Beijing_scores0,
      caption = "Forecast Accuracy for Seasonal Data",
      digits = array(4,ncol(Beijing_scores0))) %>%
  kable_styling(full_width = FALSE, position = "center") %>%
  #highlight model with lowest RMSE
  kable_styling(latex_options="striped", stripe_index = which.min(Beijing_scores0[,"RMSE"]))
```

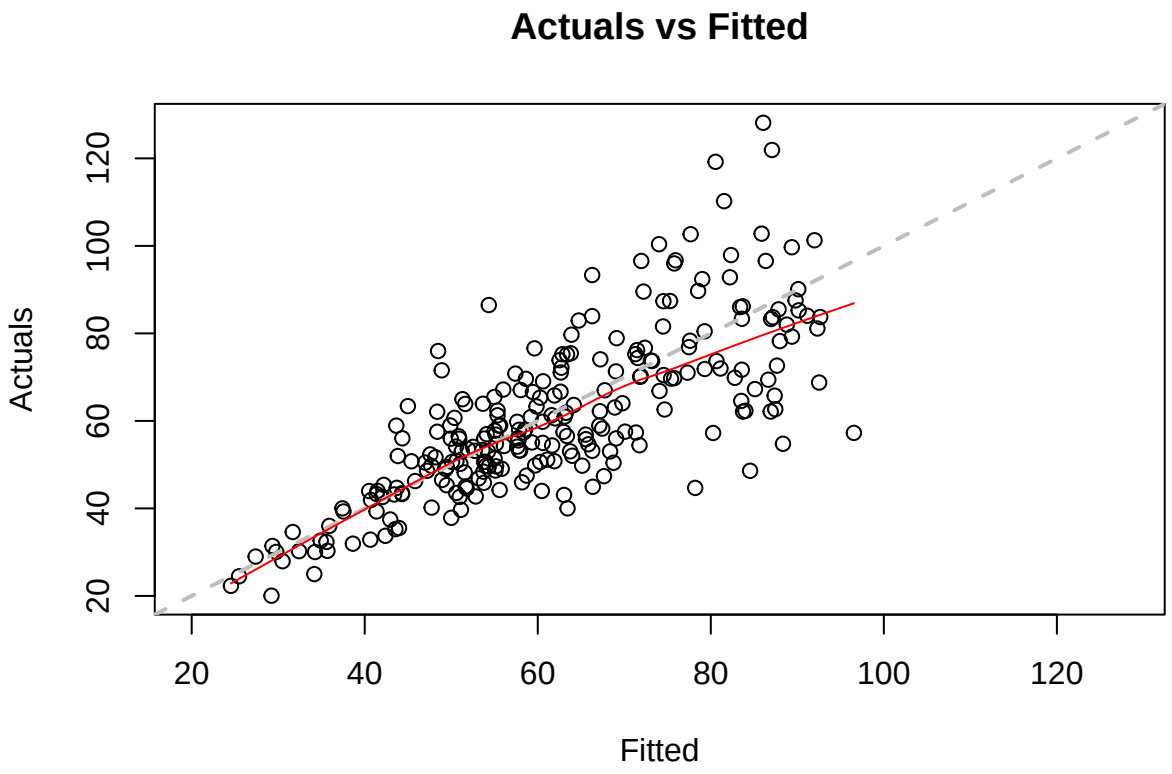
```
##save
write.csv(Beijing_scores0, row.names = FALSE,
          file = "./Data/Processed/Beijing_scores0.csv")
# 6 models table
Beijing_scores <- as.data.frame(rbind(MEAN_scores, SNAIVE_scores, SARIMA_scores,SSES_scores, M2_3_scores
row.names(Beijing_scores) <- c("MEAN", "SNAIVE","SARIMA","SSES","SNAIVE_SARIMA","SSES_SARIMA")

kbl(Beijing_scores,
      caption = "Forecast Accuracy for Seasonal Data",
      digits = array(6,ncol(Beijing_scores))) %>%
  kable_styling(full_width = FALSE, position = "center") %>%
  #highlight model with lowest RMSE
  kable_styling(latex_options="striped", stripe_index = which.min(Beijing_scores[,"RMSE"]))
```

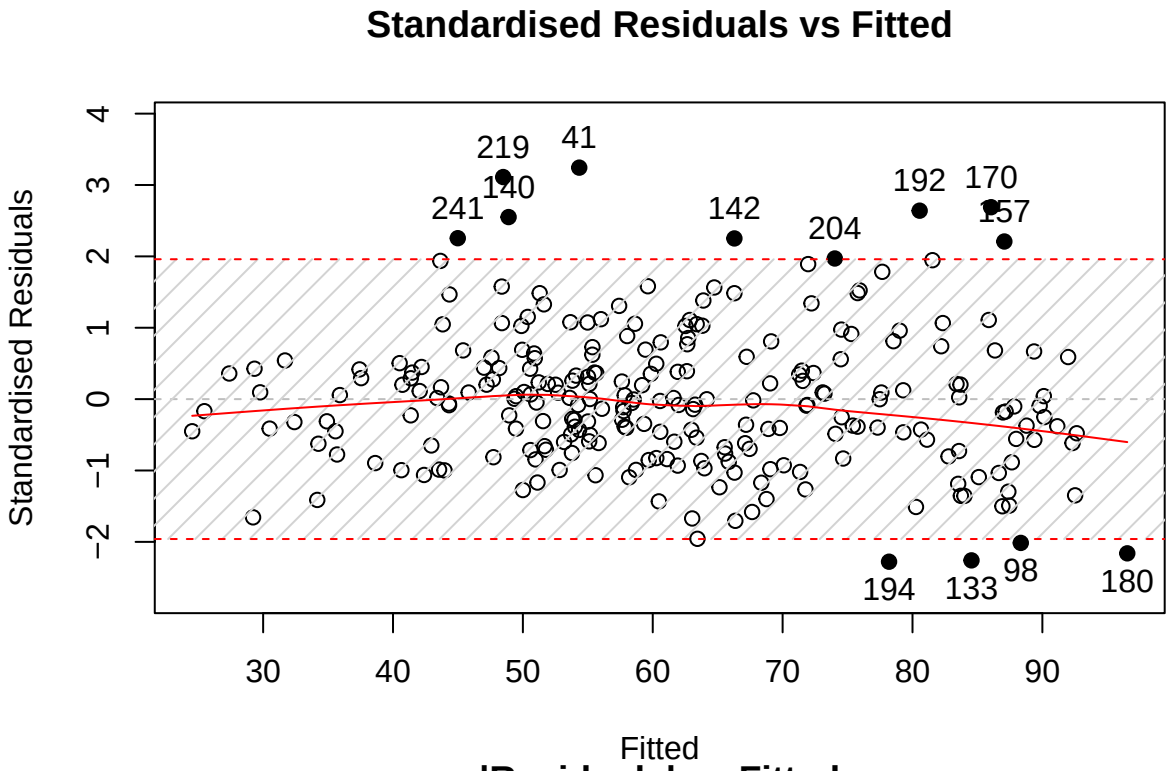
```
write.csv(Beijing_scores, row.names = FALSE,
          file = "./Data/Processed/Beijing_scores.csv")

best_model_index <- which.min(Beijing_scores[,"RMSE"])
cat("The best model by RMSE is:", row.names(Beijing_scores[best_model_index,]))
```

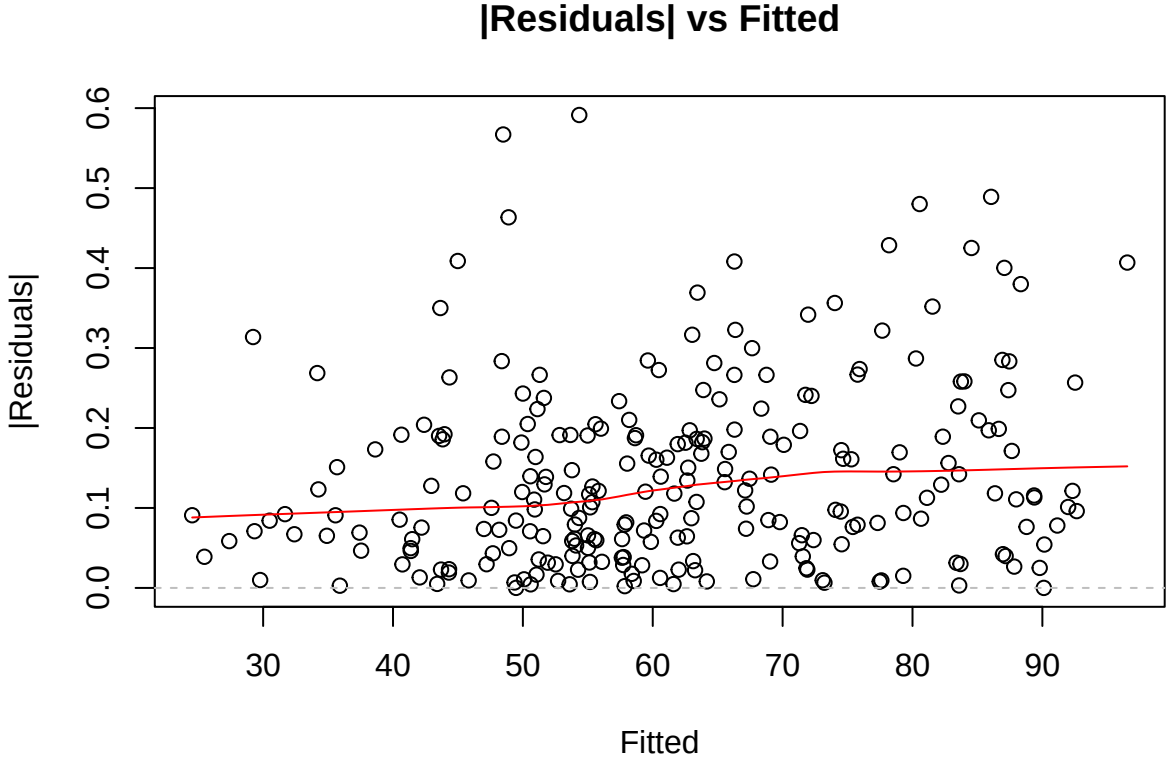
```
## The best model by RMSE is: SSES
```

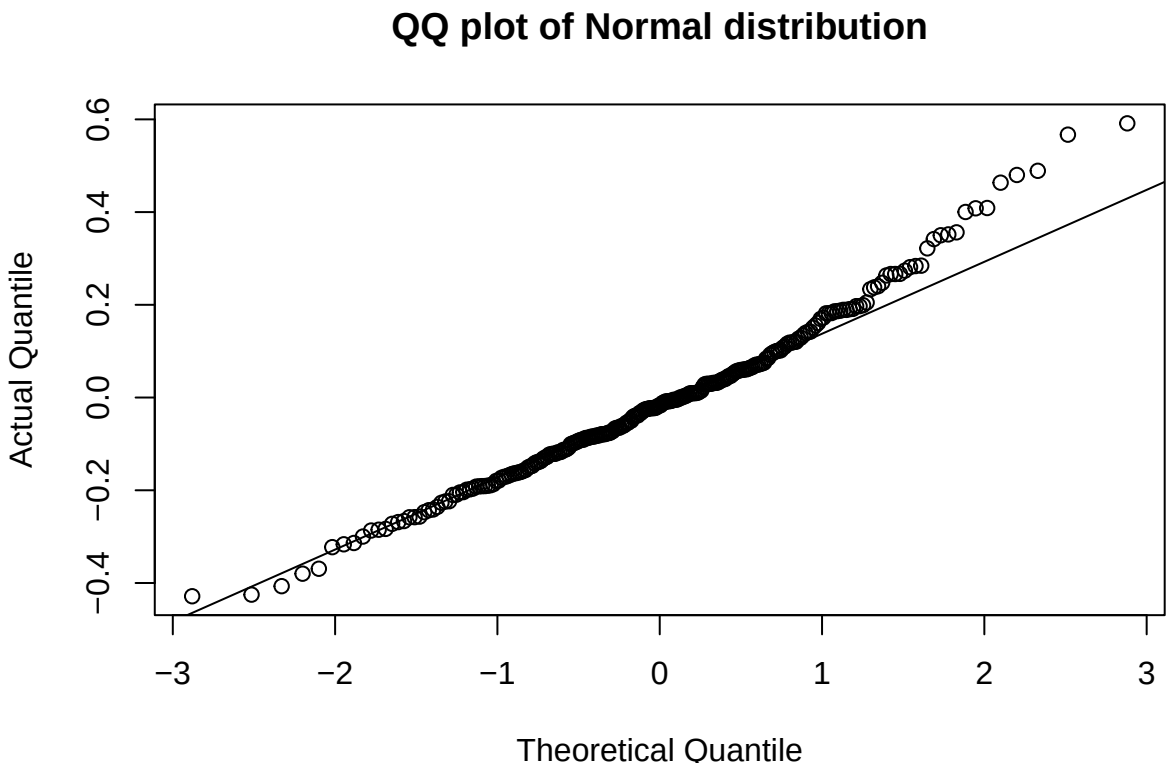
##test result-2023 ###SSES model, two year result are same

```
for_Beijing_result <- es(ts_Beijing_all,model="ZZZ",h=24,holdout=FALSE)
plot(SSES_Beijing)
```




Standardised Residuals vs Fitted
Standardised Residuals
Fitted
219
41
140
241
142
204
192
170
157
194
133
98
180

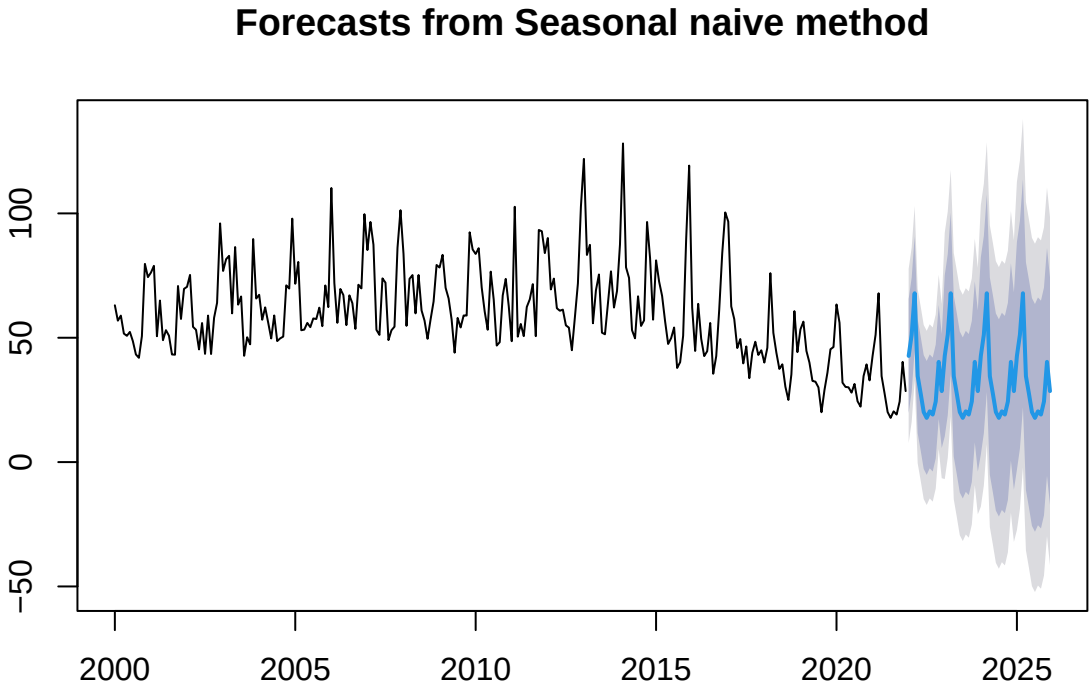

|Residuals| vs Fitted
|Residuals|
Fitted




```
forecast_Beijing <- for_Beijing_result$forecast
```

###SNAIVE same two years

```
for_Beijing_result2 <- snaive(ts_Beijing_all, h=48, holdout=FALSE)
plot(for_Beijing_result2)
```

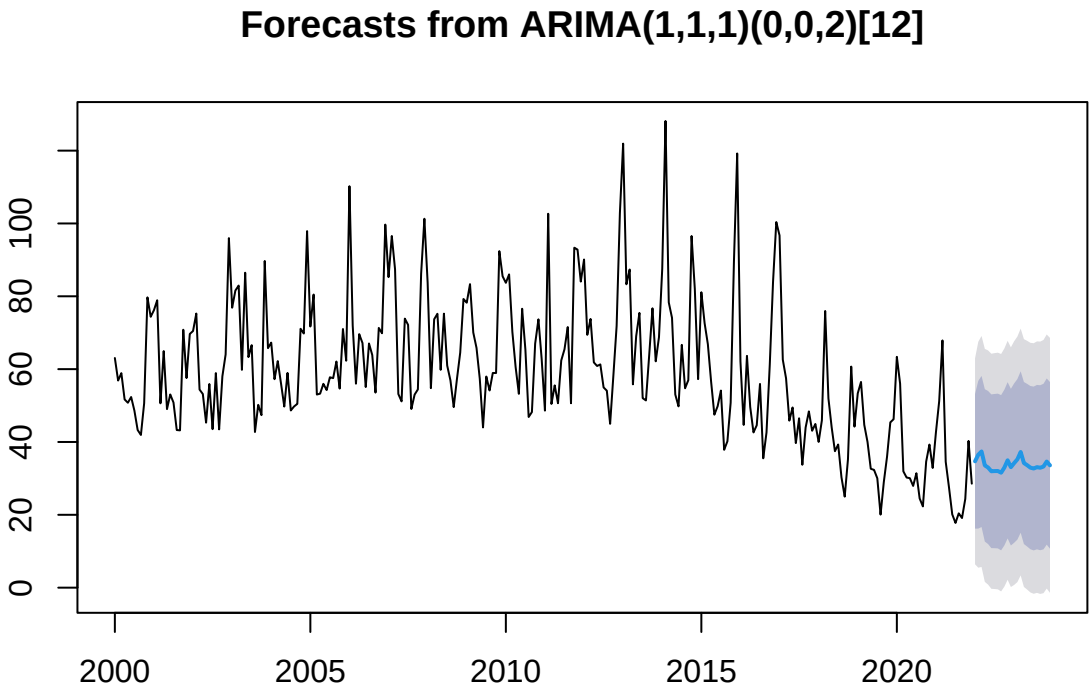


```
for_Beijing_result2$mean
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2022 42.68474 51.23041 67.88502 34.56747 27.54109 20.13027 17.76858 20.40289
## 2023 42.68474 51.23041 67.88502 34.56747 27.54109 20.13027 17.76858 20.40289
## 2024 42.68474 51.23041 67.88502 34.56747 27.54109 20.13027 17.76858 20.40289
## 2025 42.68474 51.23041 67.88502 34.56747 27.54109 20.13027 17.76858 20.40289
##           Sep      Oct      Nov      Dec
## 2022 19.11793 24.37159 40.29166 28.54776
## 2023 19.11793 24.37159 40.29166 28.54776
## 2024 19.11793 24.37159 40.29166 28.54776
## 2025 19.11793 24.37159 40.29166 28.54776
```

###SARIMA can forecast several years

```
for_Beijing_result3_fit <- auto.arima(ts_Beijing_all)
for_Beijing_result3 <- forecast(for_Beijing_result3_fit,h=24)
plot(for_Beijing_result3)
```




```
for_Beijing_result3$mean
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2022 34.68148 36.49763 37.39392 33.57311 32.98841 31.96043 32.00859 32.02408
## 2023 34.19778 35.23360 37.24690 34.21378 33.62001 32.95138 32.71680 33.09128
##           Sep      Oct      Nov      Dec
## 2022 31.56442 32.97714 34.96842 33.08234
## 2023 32.92716 33.23987 34.62669 33.57681
```

##final use SSES+SARIMA ###beijing 1100

```
#data
Beijing <- raw_wider[,c('date', 'year', 'month', 'day', '1100')] %>% rename(pm2.5 = '1100') %>% group_by
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#ts.
ts_Beijing <- ts(Beijing$monthlyPM,start=c(2000,1), frequency = 12)
#SSES
SSES_Beijing <- es(ts_Beijing,model="ZZZ",h=60,holdout=FALSE)
#SARIMA
SARIMA <- auto.arima(ts_Beijing, seasonal = TRUE)
SARIMA_Beijing <- forecast(SARIMA,h=60)
#result
R_Beijing <- (SSES_Beijing$forecast+SARIMA_Beijing$mean)/2
```

###Shanghai 3100

```
#data
Shanghai <- raw_wider[,c('date', 'year', 'month', 'day', '3100')] %>% rename(pm2.5 = '3100') %>% group_b
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#ts.
ts_Shanghai <- ts(Shanghai$monthlyPM,start=c(2000,1), frequency = 12)
#SSES
SSES_Shanghai <- es(ts_Shanghai,model="ZZZ",h=60,holdout=FALSE)
#SARIMA
SARIMA <- auto.arima(ts_Shanghai, seasonal = TRUE)
SARIMA_Shanghai <- forecast(SARIMA,h=60)
#result
R_Shanghai <- (SSES_Shanghai$forecast+SARIMA_Shanghai$mean)/2
```

###Guangzhou 4401

```
#data
Guangzhou <- raw_wider[,c('date', 'year', 'month', 'day', '4401')] %>% rename(pm2.5 = '4401') %>% group_
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#ts.
ts_Guangzhou <- ts(Guangzhou$monthlyPM,start=c(2000,1), frequency = 12)
#SSES
SSES_Guangzhou <- es(ts_Guangzhou,model="ZZZ",h=60,holdout=FALSE)
#SARIMA
SARIMA <- auto.arima(ts_Guangzhou, seasonal = TRUE)
SARIMA_Guangzhou <- forecast(SARIMA,h=60)
#result
R_Guangzhou <- (SSES_Guangzhou$forecast+SARIMA_Guangzhou$mean)/2
```

###Shenzhen 4403

```
#data
Shenzhen <- raw_wider[,c('date', 'year', 'month', 'day', '4403')] %>% rename(pm2.5 = '4403') %>% group_
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#ts.
ts_Shenzhen <- ts(Shenzhen$monthlyPM,start=c(2000,1), frequency = 12)
#SSES
SSES_Shenzhen <- es(ts_Shenzhen,model="ZZZ",h=60,holdout=FALSE)
#SARIMA
SARIMA <- auto.arima(ts_Shenzhen, seasonal = TRUE)
SARIMA_Shenzhen <- forecast(SARIMA,h=60)
#result
R_Shenzhen <- (SSES_Shenzhen$forecast+SARIMA_Shenzhen$mean)/2
```

###Tianjin 1200

```
#data
Tianjin <- raw_wider[,c('date', 'year', 'month', 'day', '1200')] %>% rename(pm2.5 = '1200') %>% group_by
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#ts.
ts_Tianjin <- ts(Tianjin$monthlyPM,start=c(2000,1), frequency = 12)
#SSES
SSES_Tianjin <- es(ts_Tianjin,model="ZZZ",h=60,holdout=FALSE)
#SARIMA
SARIMA <- auto.arima(ts_Tianjin, seasonal = TRUE)
SARIMA_Tianjin <- forecast(SARIMA,h=60)
#result
R_Tianjin <- (SSES_Tianjin$forecast+SARIMA_Tianjin$mean)/2
```

###Chongqing 1100

```
#data
Chongqing <- raw_wider[,c('date', 'year', 'month', 'day', '5000')] %>% rename(pm2.5 = '5000') %>% group_
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#ts.
ts_Chongqing <- ts(Chongqing$monthlyPM,start=c(2000,1), frequency = 12)
#SSES
SSES_Chongqing <- es(ts_Chongqing,model="ZZZ",h=60,holdout=FALSE)
#SARIMA
SARIMA <- auto.arima(ts_Chongqing, seasonal = TRUE)
SARIMA_Chongqing <- forecast(SARIMA,h=60)
#result
R_Chongqing <- (SSES_Chongqing$forecast+SARIMA_Chongqing$mean)/2
```

###Chengdu 5101

```
#data
Chengdu <- raw_wider[,c('date', 'year', 'month', 'day', '5101')] %>% rename(pm2.5 = '5101') %>% group_by
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```
#ts.
ts_Chengdu <- ts(Chengdu$monthlyPM,start=c(5101,1), frequency = 12)
#SSES
SSES_Chengdu <- es(ts_Chengdu,model="ZZZ",h=60,holdout=FALSE)
#SARIMA
SARIMA <- auto.arima(ts_Chengdu, seasonal = TRUE)
SARIMA_Chengdu <- forecast(SARIMA,h=60)
#result
R_Chengdu <- (SSES_Chengdu$forecast+SARIMA_Chengdu$mean)/2
```

###all

```
all_forcast <- data.frame(R_Beijing, R_Shanghai, R_Guangzhou, R_Shenzhen, R_Tianjin, R_Chongqing, R_Chen
date_sequence <- seq(from = as.Date("2022-01-01"), by = "month", length.out = 60)
all_forcast$Date <- date_sequence
all_forcast <- all_forcast[,c(7,1:6)]
write.csv(all_forcast, row.names = FALSE,
          file = "./Data/Processed/all_forcast.csv")
```

#seasonal+trend

```
stl_Beijing <- stl(ts_Beijing, s.window = "periodic")
stl_Shanghai <- stl(ts_Shanghai, s.window = "periodic")
stl_Guangzhou <- stl(ts_Guangzhou, s.window = "periodic")
stl_Shenzhen <- stl(ts_Shenzhen, s.window = "periodic")
stl_Tianjin <- stl(ts_Tianjin, s.window = "periodic")
stl_Chongqing <- stl(ts_Chongqing, s.window = "periodic")
stl_Chengdu <- stl(ts_Chengdu, s.window = "periodic")

seasonal_Beijing <- stl_Beijing$time.series[c(1:12), "seasonal"]
seasonal_Shanghai <- stl_Shanghai$time.series[c(1:12), "seasonal"]
seasonal_Guangzhou <- stl_Guangzhou$time.series[c(1:12), "seasonal"]
seasonal_Shenzhen <- stl_Shenzhen$time.series[c(1:12), "seasonal"]
seasonal_Tianjin <- stl_Tianjin$time.series[c(1:12), "seasonal"]
seasonal_Chongqing <- stl_Chongqing$time.series[c(1:12), "seasonal"]
seasonal_Chengdu <- stl_Chengdu$time.series[c(1:12), "seasonal"]

trend_Beijing <- stl_Beijing$time.series[, "trend"]
trend_Shanghai <- stl_Shanghai$time.series[, "trend"]
trend_Guangzhou <- stl_Guangzhou$time.series[, "trend"]
trend_Shenzhen <- stl_Shenzhen$time.series[, "trend"]
trend_Tianjin <- stl_Tianjin$time.series[, "trend"]
trend_Chongqing <- stl_Chongqing$time.series[, "trend"]
trend_Chengdu <- stl_Chengdu$time.series[, "trend"]
```

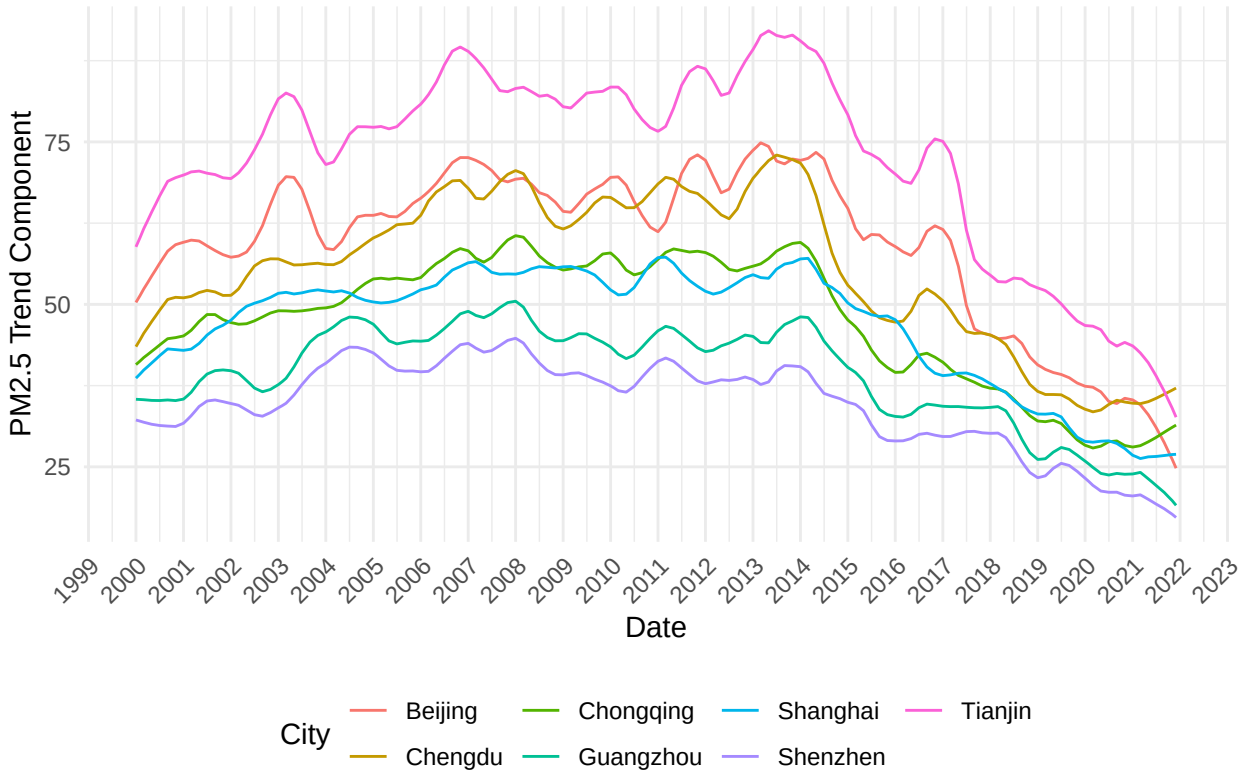
##seasonalALL

```
seasonal_data <- data.frame(
  city = rep(c("Beijing", "Shanghai", "Guangzhou", "Shenzhen", "Tianjin", "Chongqing", "Chengdu"), each
  month = rep(month.abb, 7),
  seasonal = c(seasonal_Beijing, seasonal_Shanghai, seasonal_Guangzhou, seasonal_Shenzhen, seasonal_Tia
)
#plot1_seasonal,"not good
plot1 <- ggplot(seasonal_data, aes(x = month, y = seasonal, group = city)) +
  geom_line() +
  scale_x_discrete(limits = month.abb) +
  facet_wrap(~city, nrow = 3) +
  labs(title = "Seasonal Patterns in 7 Chinese Cities", x = "Month", y = "Seasonal Component") +
  theme_minimal()

#plot2_seasonal,
plot2 <- ggplot(seasonal_data, aes(x = month, y = seasonal, group = city, colour = city)) +
  geom_line() +
  scale_x_discrete(limits = month.abb) +
  labs(title = "Figure 2. Seasonality in PM2.5 in 7 Chinese Cities", x = "Month", y = "PM2.5 Seasonal C
  theme_minimal() +
  theme(legend.position = "bottom") +
  guides(colour = guide_legend(title = "City"))


write.csv(seasonal_data, row.names = FALSE,
          file = "./Data/Processed/seasonal_data.csv")
```

##trendALL

```
date_seq <- seq(as.Date("2000-01-01"), as.Date("2021-12-01"), by = "month")
trend_data <- data.frame(
  city = rep(c("Beijing", "Shanghai", "Guangzhou", "Shenzhen", "Tianjin", "Chongqing", "Chengdu"), each
  date = rep(date_seq, 7),
  trend = c(trend_Beijing, trend_Shanghai, trend_Guangzhou, trend_Shenzhen, trend_Tianjin, trend_Chongq
)

ggplot(trend_data, aes(x = date, y = trend, group = city, colour = city)) +
  geom_line() +
  labs(title = "Figure 3. Trends in PM2.5 in 7 Chinese Cities", x = "Date", y = "PM2.5 Trend Component"
  scale_x_date(breaks = "1 year", date_labels = "%Y") +
  theme_minimal() +
  theme(legend.position = "bottom",
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  guides(colour = guide_legend(title = "City"))
```

Figure 3. Trends in PM2.5 in 7 Chinese Cities




```
write.csv(trend_data, row.names = FALSE,
          file = "./Data/Processed/trend_data.csv")
```