STAT 443 Spring 2024 Final Project Report

Daolin(Chase) An

UW ID: 20885166

Contents

1 Scenario 1: Hydrological Forecast

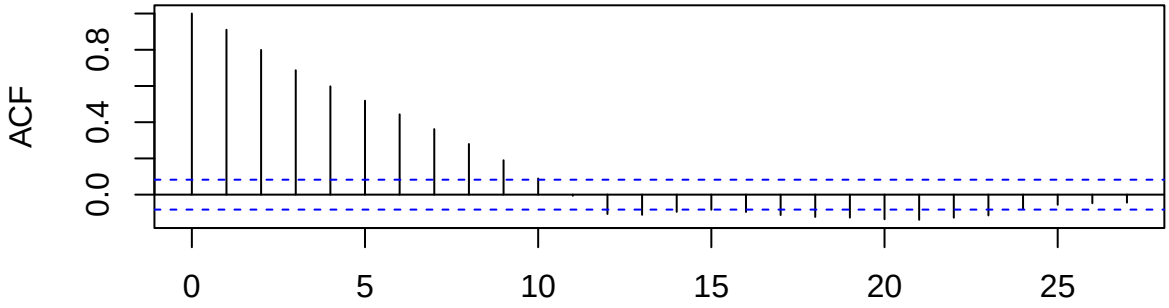
1.1 Analysis

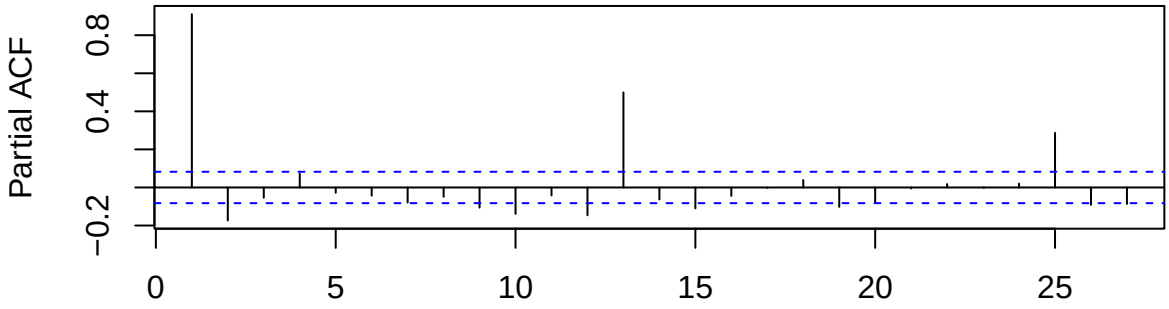
I first plot the original series, and observe that:

- There is high variability in the first half of the series but then the variability becomes more moderate. This may indicate that the original time series is non-stationary.
- The data also suggests a possible seasonal pattern, as it shows regular and frequent peaks and troughs.
- The data covers 576 consecutive months, representing a 48-year period. Within each year, the value of data typically increase from January(assume the first value was recorded in January) to May, peaking at May, and then it decreases from May to December.

Then I apply seasonal differencing on the data to make it more stationary and take a log transformation to address the unstable variability. I think a 12th-order difference is appropriate in this case(yearly).

Next, I visually check acf/pacf of the differenced data and perform KPSS test on it. Visual inspection and KPSS result indicate that the differenced data is stationary, so I begin to fit a model.





Seasonal Component: It appears that at the seasons, the PACF is cutting off at lag 1s(s = 12). The ACF is tailing off at at lag 1s, 2s. These results implies an SAR(1), SMA(1), P = 1, Q = 1, in the season(s = 12).

Non-Seasonal Component: Inspecting the sample ACF and PACF at the lower lags, it appears as though both are tailing off. This suggests an ARMA(1, 1) within the seasons, p = q = 1. Now since both the acf and pacf of differenced data tails off, and the time series shows a general seasonal pattern, I start with a SARIMA$(1,1,1) \times (1,1,1)_{12}$ model.

I generated several models, and these models are compared via AIC/BIC, expanding window cross-validation error(see Appendix), and residual analysis.

I end up with a SARIMA$(3,1,1) \times (1,1,1)_{12}$ model with AIC=-4.223715, BIC=-4.169838, Expanding Window Cross-Validation MSE = 3.767112. I do not pick the other models because they either show higher AIC/BIC/CV_MSE or show more severe violations to model assumptions like white noise, normality. (Details can be found in my Supplementary file)

The figure below gives the diagnostic plots for my final model. The model seems to capture all the autocorrelation with only one lag outside the bands, so it is not a big concern. The residuals are randomly scattered around y=0, and there is one part of noticeable spike in the middle part which can be seen as outlier and it would not affect the model's overall performance. Moreover, the Box-Ljeung-Pierce test supports the assumption of white noise.

One potential problem is that although the model is approximately normal, it has a slightly heavier tail which may negatively affect the performance of our prediction interval. It may be more likely to observe extreme values in this case, so the prediction interval may be adjusted wider to cover them.

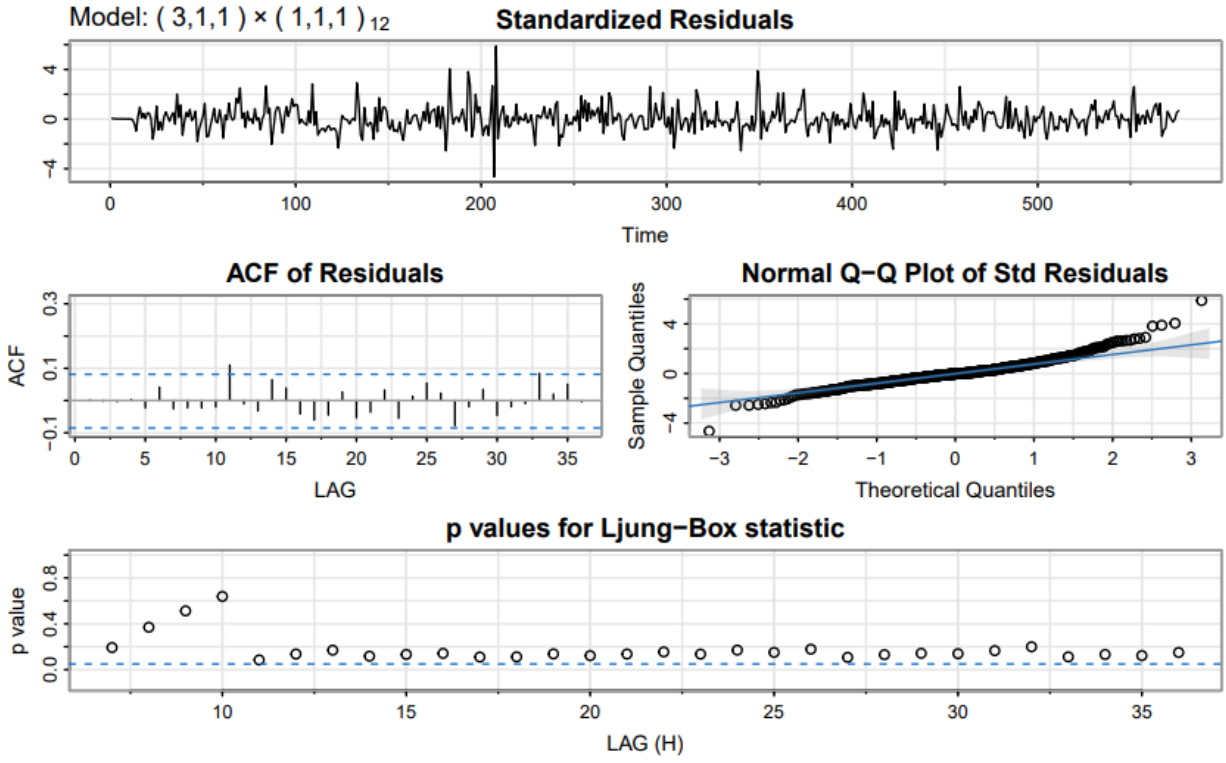
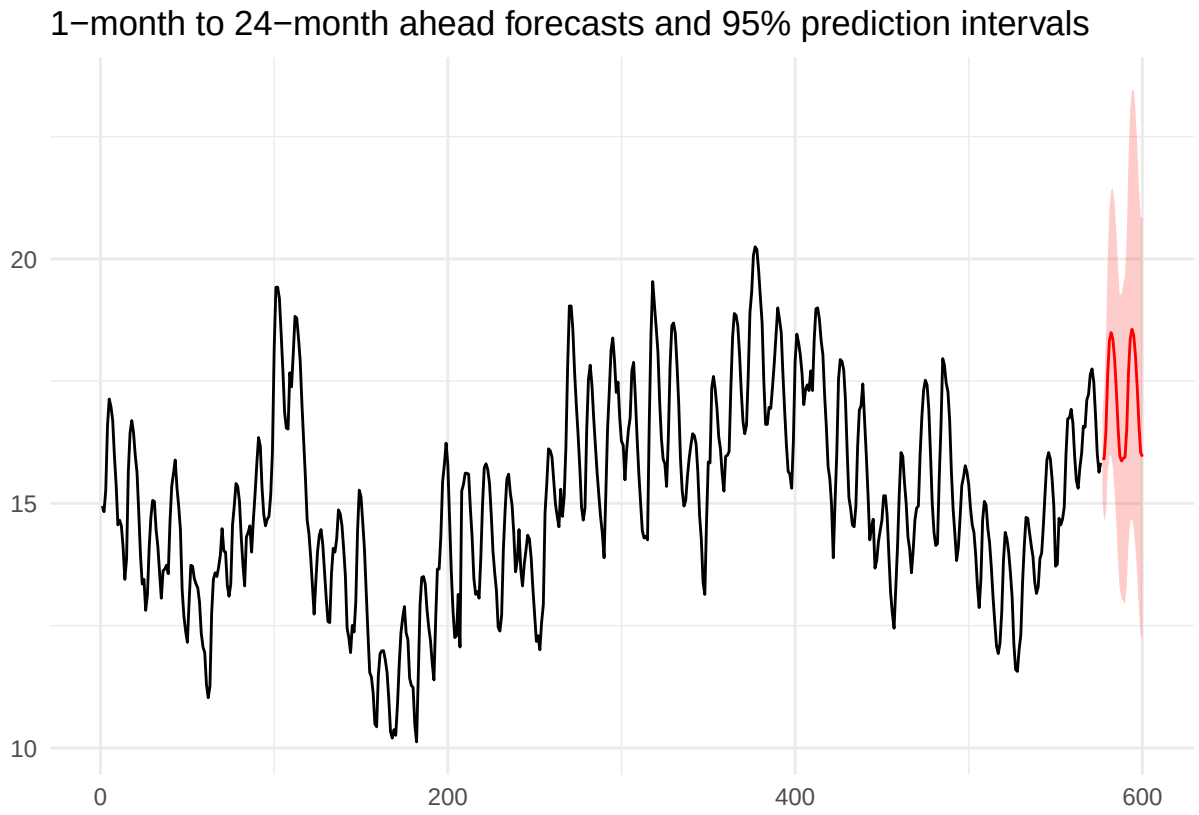


Figure 1: Diagnostics plots for SARIMA(3, 1, 1) × (1, 1, 1)12 model on logged data

1.2 Forecasts and 95% prediction intervals



2 Scenario 2: Financial Risk Forecast

2.1 Analysis

Since the series are all "financial", I focus on GARCH-type models in this scenario. I manually fit several models to each stock, compare them, and pick the most reasonable model among them. The same process is followed to analyze all the stock data, so I will just present one typical case.

We are given the log differenced data, so stationarity is not a concern to me (KPSS test and visual inspection are used to confirm stationarity). Instead, I mainly check:

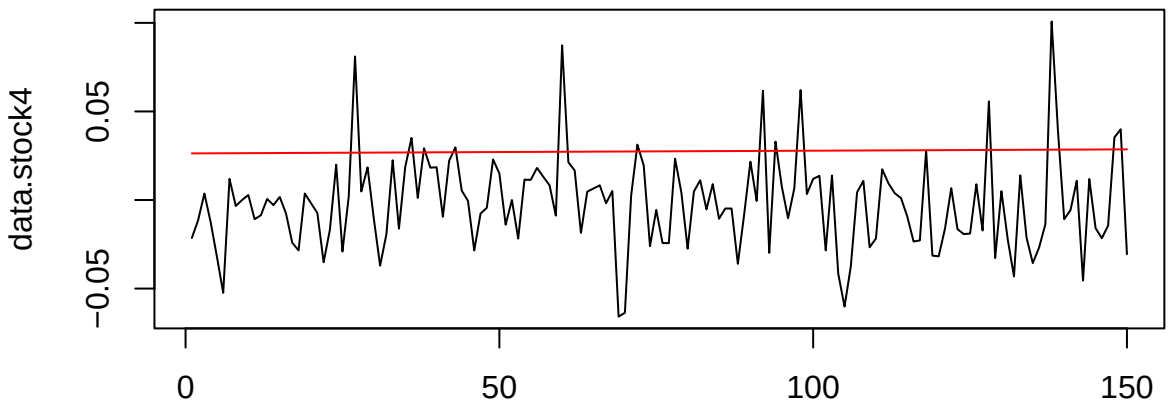
- the squared time series to see if there are significant volatility clustering
- the acf of squared time series to confirm volatility clustering and identify lags that show significant autocorrelations

After detecting volatility clustering, I fit models and usually start with a GARCH(1, 1) model. I then check its validity by overlay its estimated conditional variance on the original series to see if the model can capture the structure/volatility in the series. The model's residuals are also checked to ensure it satisfies assumptions(a sequence of with mean 1, no pattern) of GARCH-type model.

For most of the given data, GARCH(1, 1) would be way too under-specified. To address this, I typically:

- pick higher orders based on AIC/BIC/Expanding Window Cross-Validation MSE or
- use a more complex model such as "eGARCH"

For example, a GARCH(1, 1) model of stock4

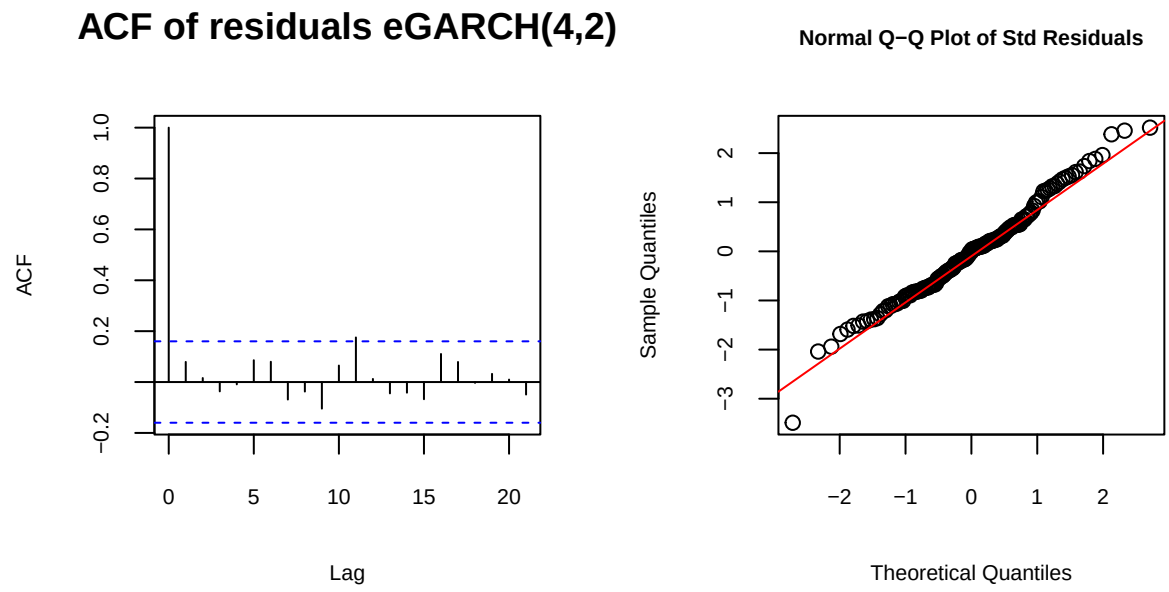


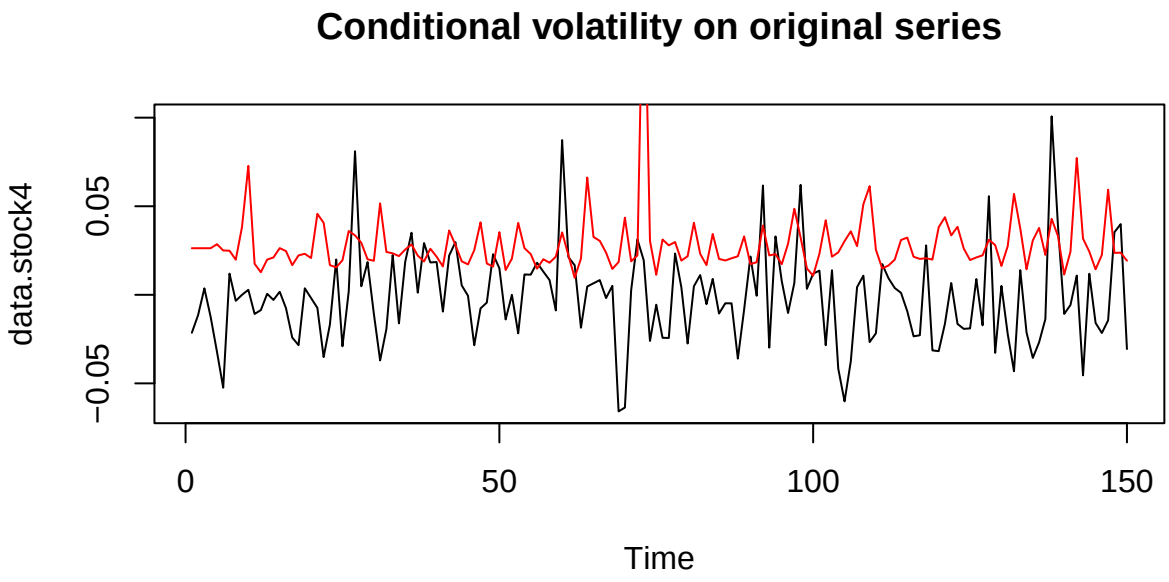
It is clear the model does not capture the pattern in the data which may indicate the number of parameters included is not enough or model is too simple.

After adding number of parameters and using more complex models, compared by information criterion AIC/BIC and cross-validation MSE, I end up with a eGARCH(4, 2) model with AIC=-4.389625, BIC=-4.148775, CV MSE= 0.0009684144. The mean of the residuals is $0.9613752 \approx 1$. (Model comparison and details can be found in my supplementary file)

The plots below gives the diagnostic plots of my final model for stock4. The model seems to capture all

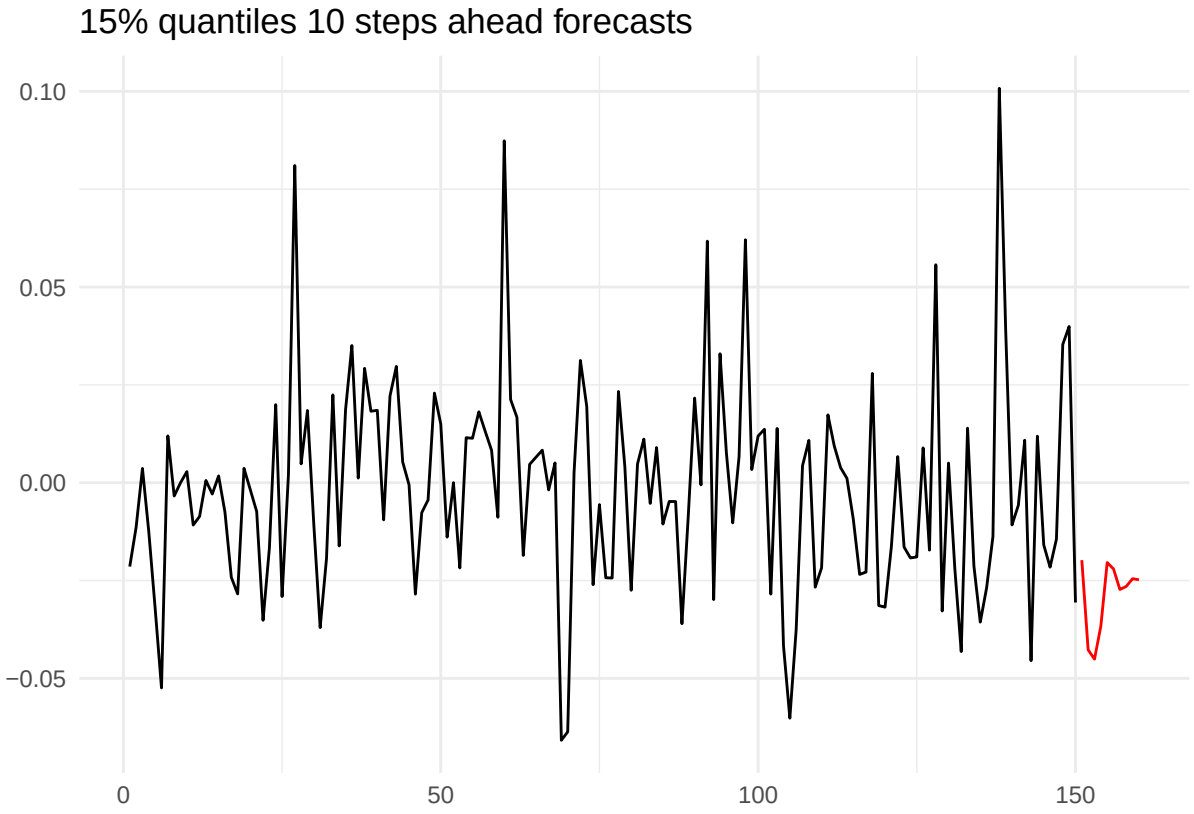
the autocorrelation with only one lag outside the bands, which is a good sign. The standardized residuals approximately follow a normal distribution. The conditional volatility laid on the original times series seem to do a good job at capturing the variablity in the data.





2.2 15% quantiles 10 steps ahead forecasts for stock4, and plot

```
##  [1] -0.01979917 -0.04271685 -0.04507417 -0.03662864 -0.02040298 -0.02205253
##  [7] -0.02728304 -0.02650030 -0.02452623 -0.02479479
```


15% quantiles 10 steps ahead forecasts
0.10
0.05
0.00
−0.05
0
50
100
150

3 Scenarios 3 and 4: Imputation and Multivariate Time Series Forecasting

3.1 Analysis

The beer data covers 435 consecutive months, representing a roughly 36 year period, however the values between row 200 and row 230 are missing. There seems to be no observable seasonal pattern by purely visual inspection.

Of all the extra data provided, I hypothesize temperature likely has a significant impact on the beer consumption based on common sense. This hypothesis is confirmed later by my model summary. The temperature data covers a broader range of dates than the beer data, which makes it easier to align their dates and values for use of multivariate time series models. I do not observe any direct relationship between beer consumption and other extra data like car data, electricity data. Therefore, I would not include these as exogenous variables to fit my model.

I initially split the beer data into two parts and analyze each part separately.

First part(corresponds to values from row1 to row200)

- There is a clear increasing trend and variability shown by its time series plot.
- There also appears to be some seasonality with repeating patterns at regular intervals(12 month by visual inspection).
- The ACF show a slow decay, and significant autocorrelations occur at many lags.

Second part(corresponds to values from row231 to the last row)

- There is no observable trend and seasonality for the second part.
- The variability seems stable with minor fluctuations.
- The ACF shows a quick decay at lower lags, but it exhibits a recurring pattern.

Based on the findings I:

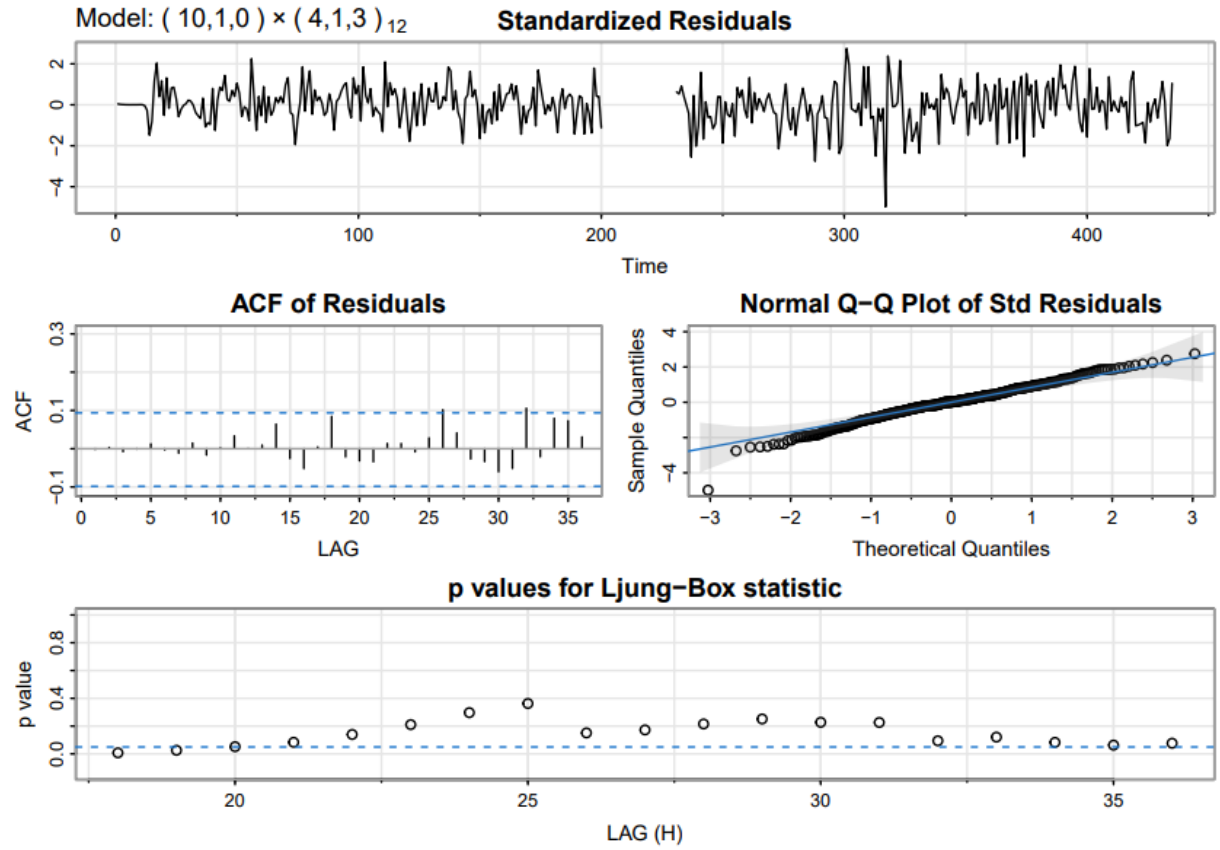
- take a log transformation overall to reduce the variability in the first part
- apply seasonal differencing with a lag of 12 to address non-stationarity

After these are done, I proceed to fit models. It is hard to pick seasonal or non-seasonal components orders by inspection since the data is split and contains missing values. Therefore, I conduct a grid-search over possible parameters and select a combination of orders that yields a reasonably low AIC/BIC. The optimal model I find is SARIMA$(10,1,0) \times (4,1,3)_{12}$ with temperature as an exogenous variable.

Then I pass this model to KalmanSmooth() to impute the missing values.

After imputing the missing values, I try different order combinations to see if there are better models. It turns out none improve my model, so I just stick with it for forecasting. Since there is no further temperature data available, I use its mean as the value for argument newexg.

The figure below gives the diagnostic plots for my final model. The model captures all the known autocorrelation with only two lags outside the bands, so it is generally good. The residuals are randomly scattered around y=0 and the model looks approximately normal. Moreover, the Box-Ljeung-Pierce test supports the assumption of white noise although two points are below the confidence band(expected to see 5% of time).

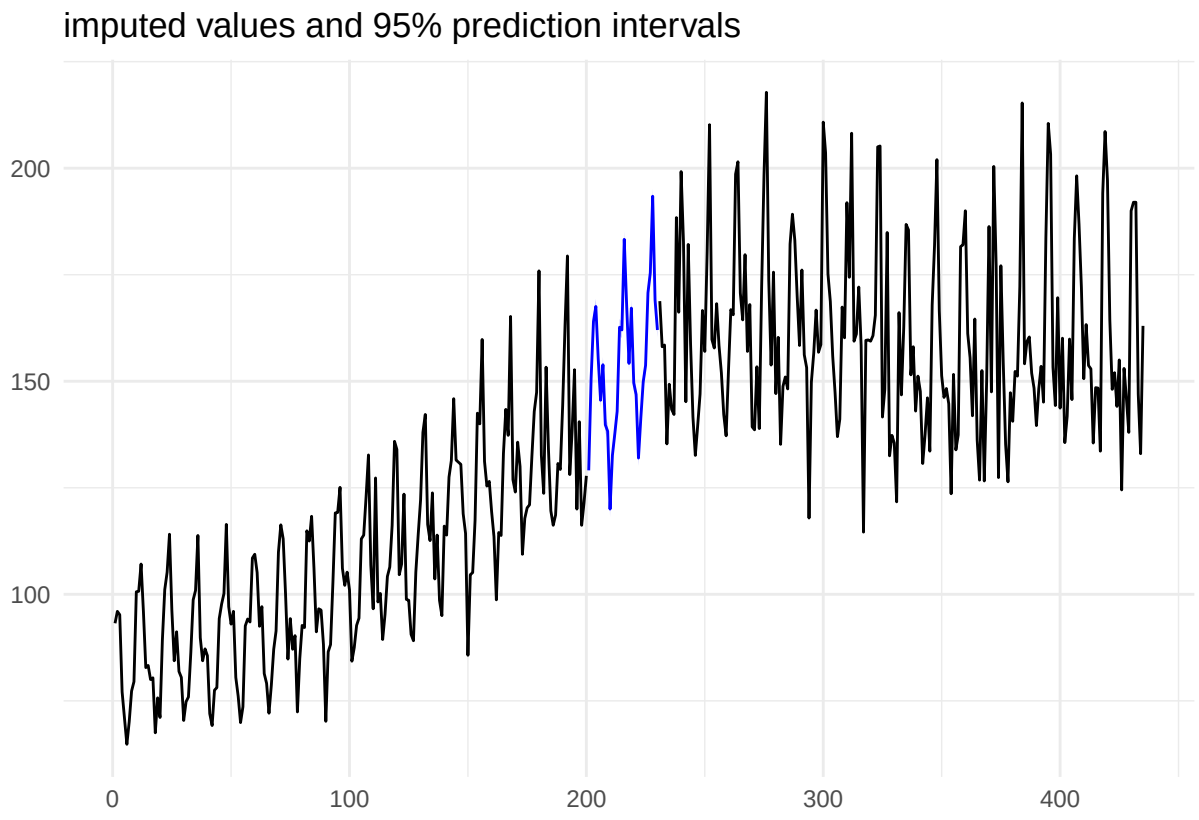


3.2 Imputations and 95% prediction intervals

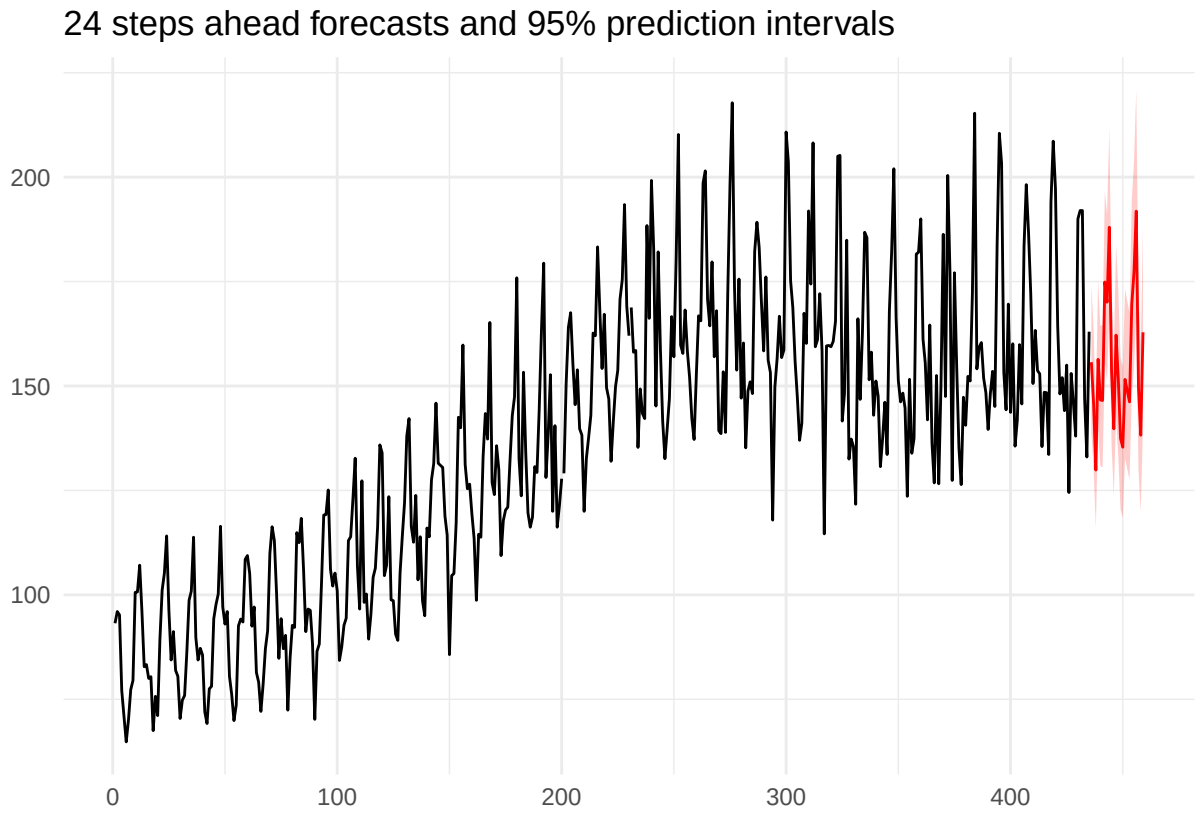
Note the prediction intervals are too narrow so it may be hard to see. I will present the 95% lower/upper prediction bounds then.

```
##  [1] 127.2759 148.7719 162.2387 165.7353 153.5163 143.7187 152.0447 137.9408
##  [9] 136.4058 118.1387 130.7667 135.7183 141.0370 160.7536 160.1206 181.3791
## [17] 166.0602 152.2724 165.2582 147.7207 145.0260 130.1282 139.5820 148.0236
## [25] 151.9088 168.9398 173.6385 191.6034 167.2885 160.2217
```

```
##  [1] 130.9045 152.3996 165.8664 169.3632 157.1435 147.3688 155.7023 141.6013
##  [9] 140.0949 121.8725 134.4912 139.5297 144.8835 164.5864 163.9539 185.2127
## [17] 169.8907 156.1167 169.0692 151.4469 148.7643 133.8221 143.2480 151.6893
## [25] 155.5664 172.5745 177.2716 195.2359 170.9154 163.8493
```


imputed values and 95% prediction intervals
200
150
100
0
100
200
300
400

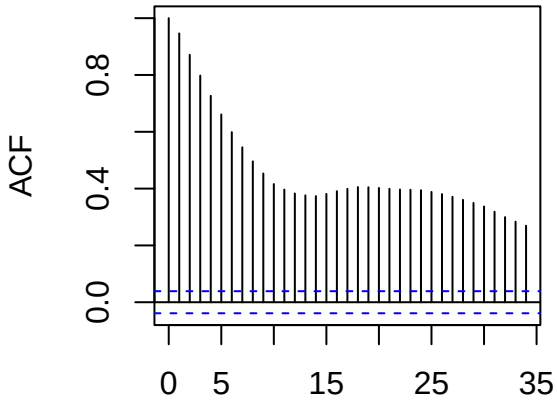
3.3 Forecasts and 95% prediction intervals

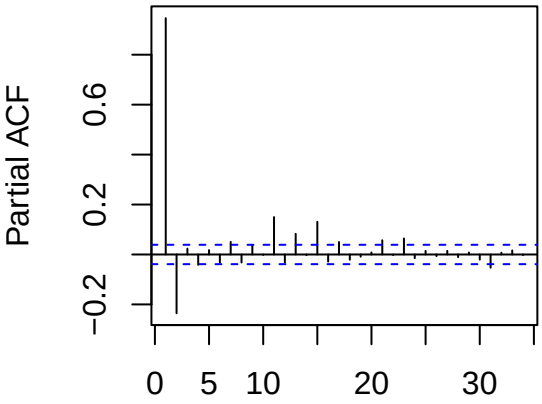


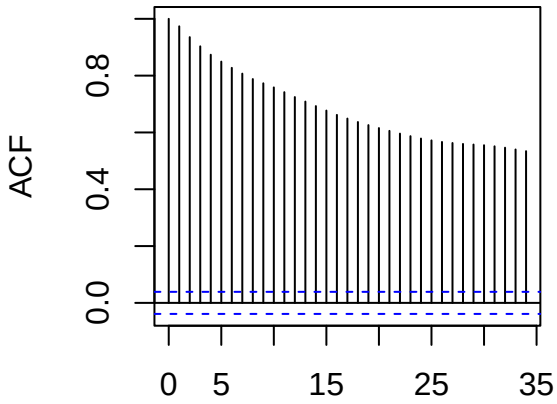
4 Scenarios 5: Long Horizon Pollution Forecasting

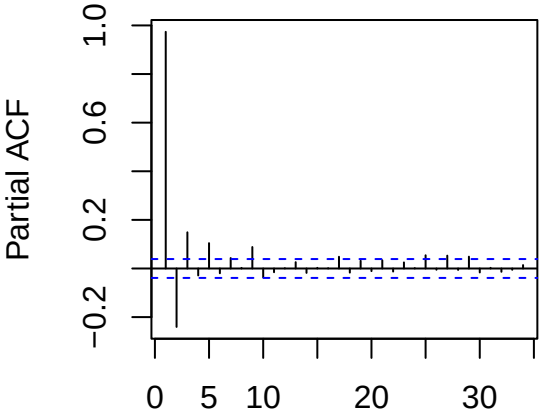
4.1 Analysis

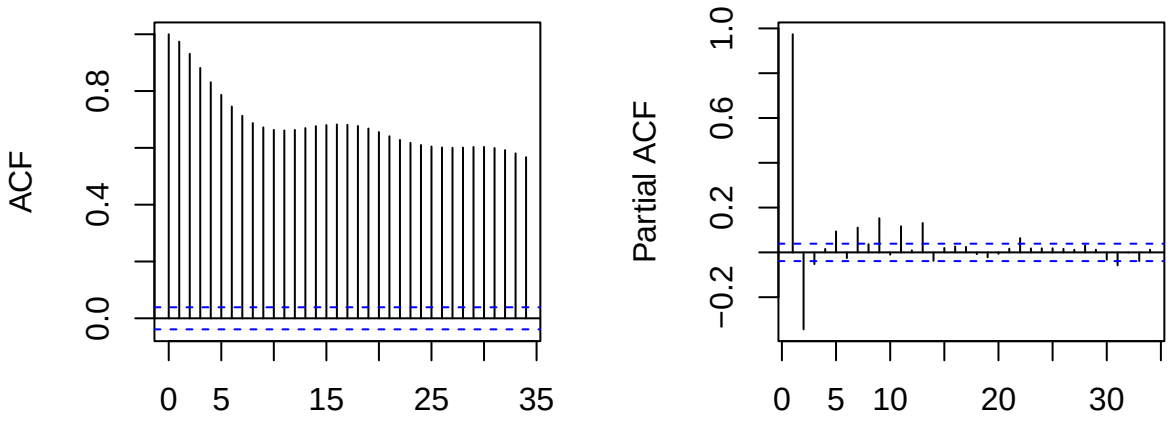
Initially, I plot the acf/pacf for each pollution series. I found that all the series follow some kind of ARMA process:











Therefore, I begin with some VARMA models. I have tried a few combination of orders, but they are not satisfying shown by model diagnostics.

Then I decide to fit models for each series instead of using multivariate methods. For each series:

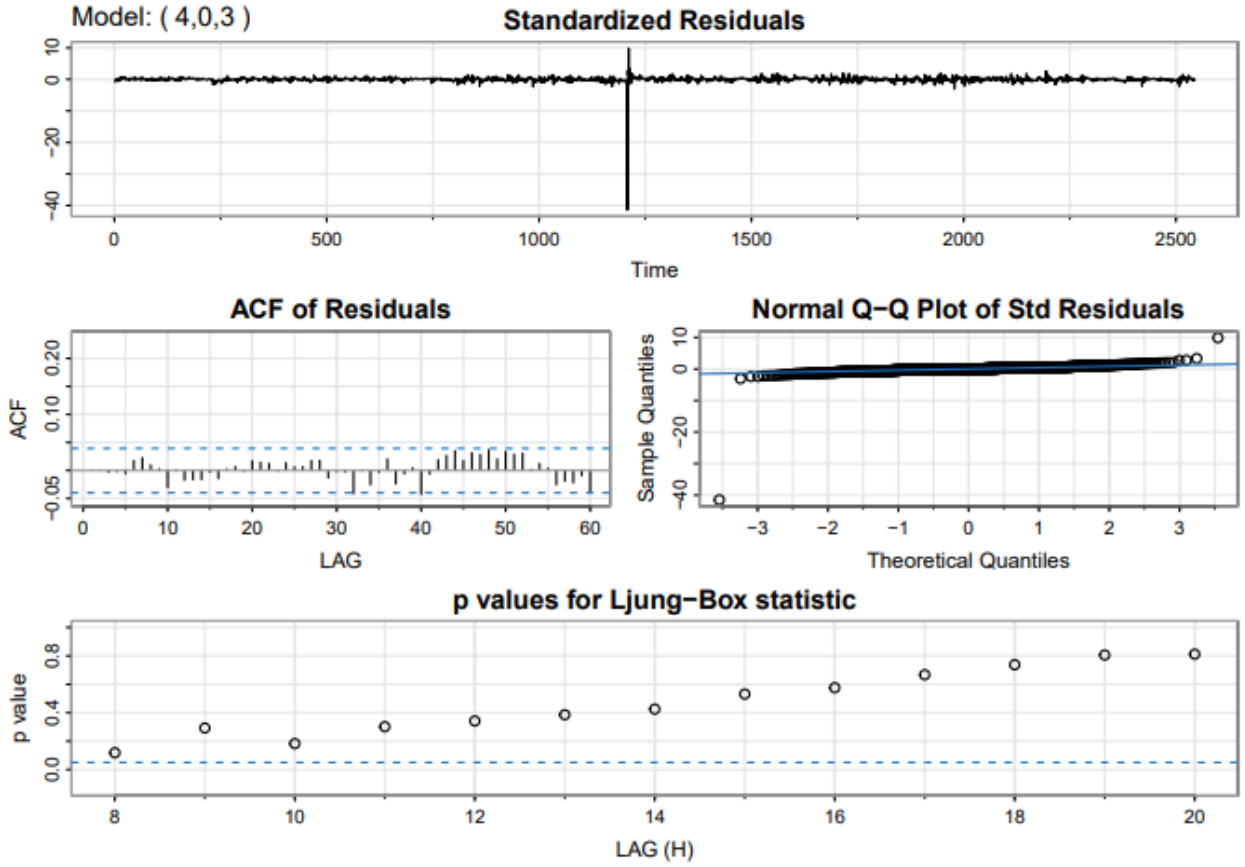
- Auto.arima() is used to pick starting orders.
- Ordering picking are based on AIC/BIC and analysis on the residuals.

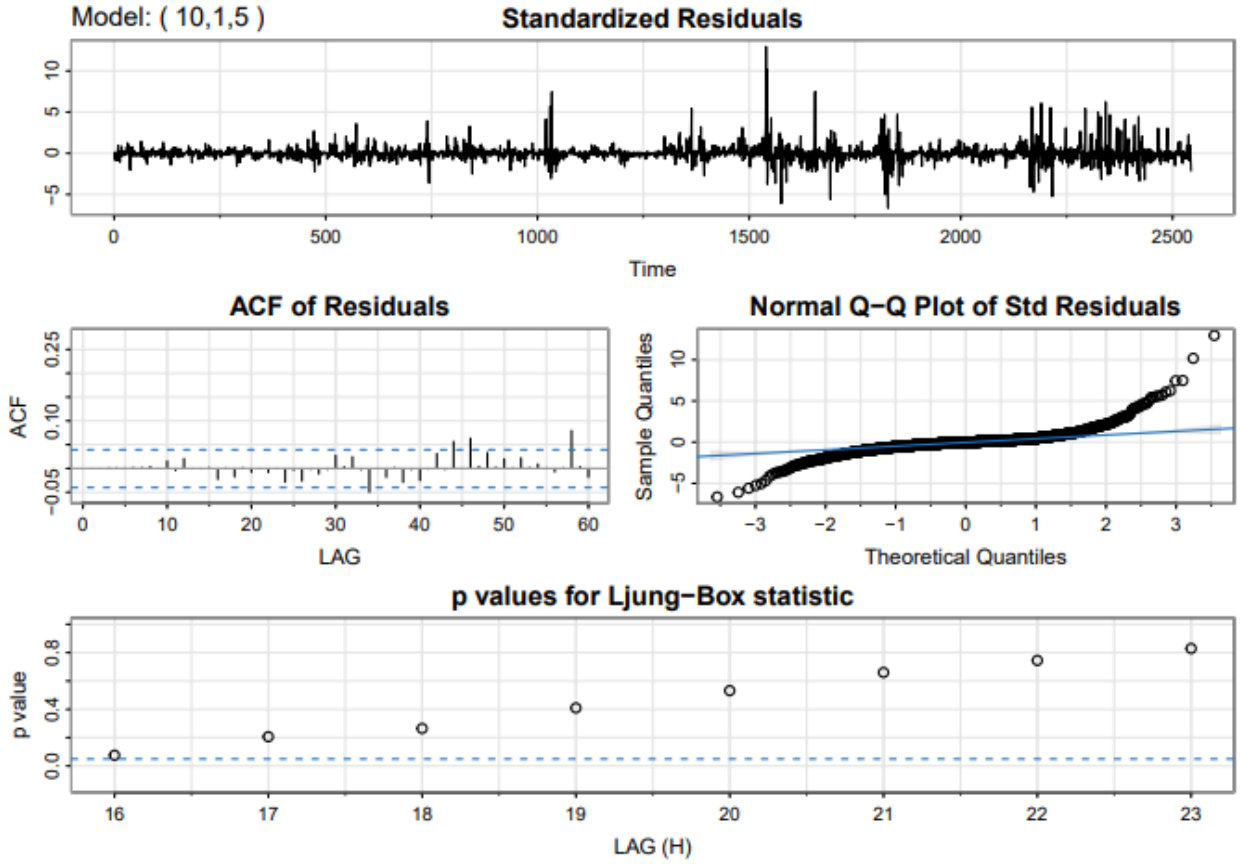
A log transformation is taken to stabilized the high variability for pollutionCity1 series.

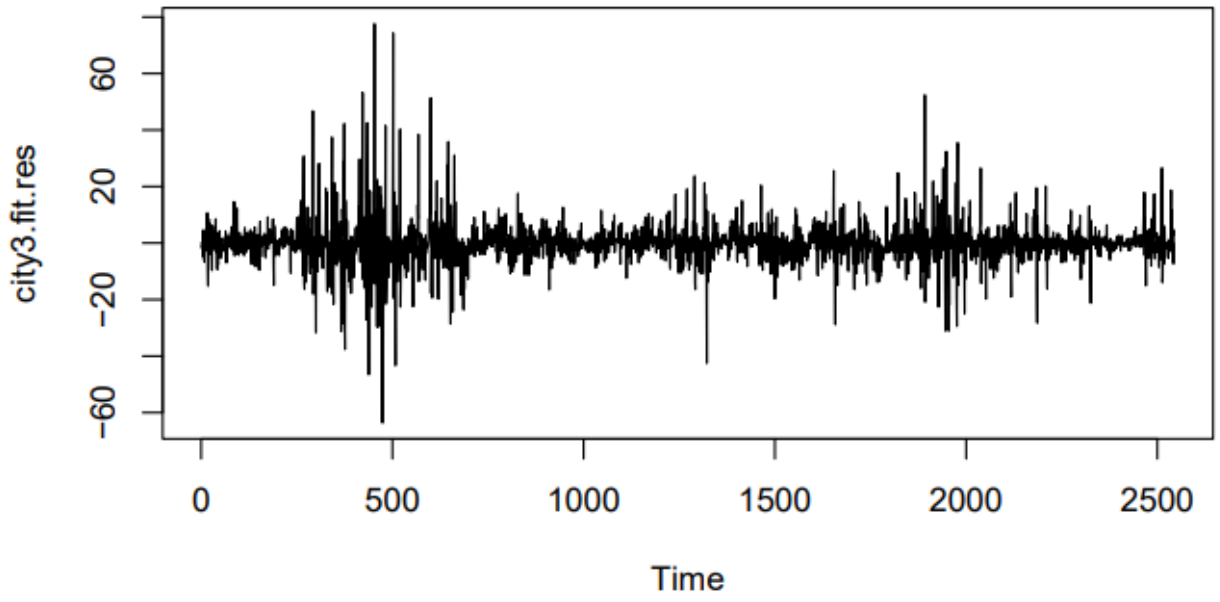
PollutionCity2 and PollutionCity3 series both show minor volatility clustering by visual inspection, so I consider ARMA-GARCH models for them. However, the ARMA-GARCH models are not satisfying, and do not improve the pure ARMA model. (Details can be found in supplementary files).

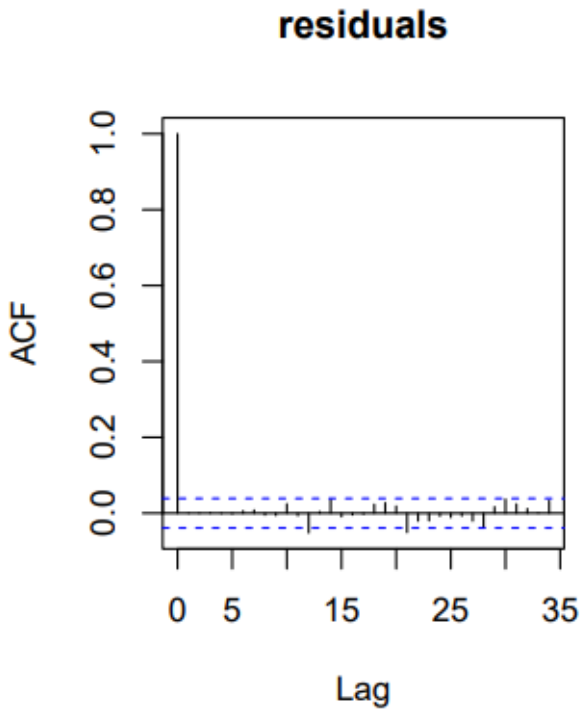
I end up with one pure ARMA model for each pollution series.

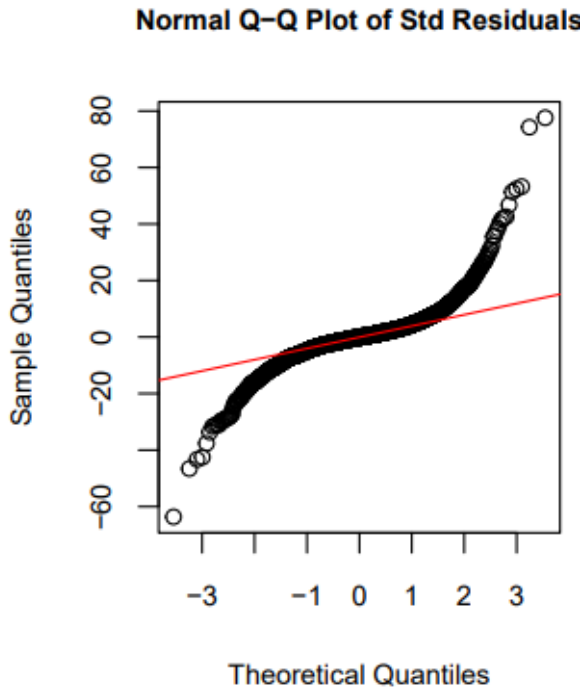
- For pollution series1, the model is ARMA(4, 0, 3). The residuals from the model resemble white noise, tested using Ljung−Box statistic. The model is also approximately normal, confirmed by Normal Q-Q test. There is also no autocorrelation(pattern) remaining in the residuals.
- For pollution series2 and pollution series3, the final models are ARMA(10, 1, 5) and ARMA(8, 1, 3) respectively. Residuals from both models show no autocorrelations, and resemble white noise. Both of them have heavier tails, so the corresponding prediction intervals probably become wider to cover extreme values(heavier tails).

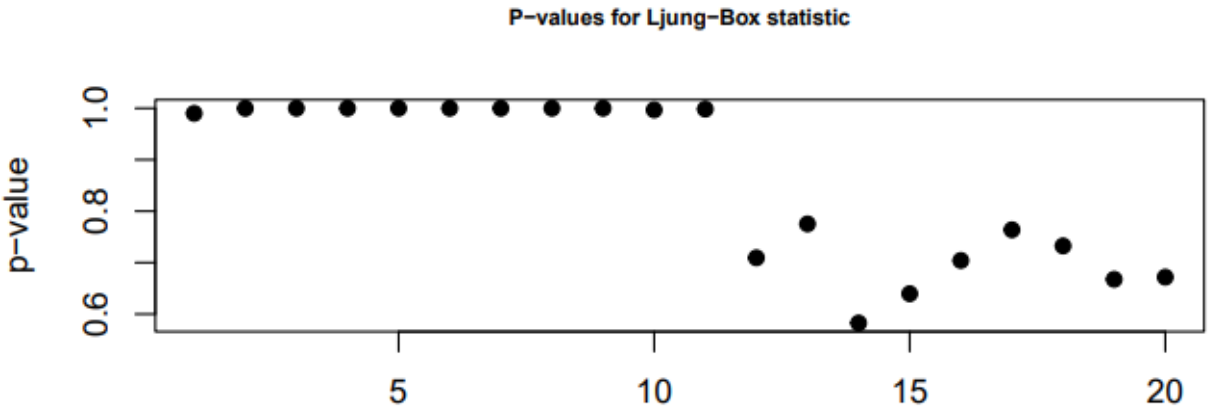

Model: (4,0,3)
Standardized Residuals
Time
ACF of Residuals
ACF
LAG
Normal Q−Q Plot of Std Residuals
Sample Quantiles
Theoretical Quantiles
p values for Ljung−Box statistic
p value
LAG (H)


Model: (10,1,5)
Standardized Residuals
Time
ACF of Residuals
ACF
LAG
Normal Q−Q Plot of Std Residuals
Sample Quantiles
Theoretical Quantiles
p values for Ljung−Box statistic
p value
LAG (H)


city3.fit.res
60
20
−20
−60
0
500
1000
1500
2000
2500
Time

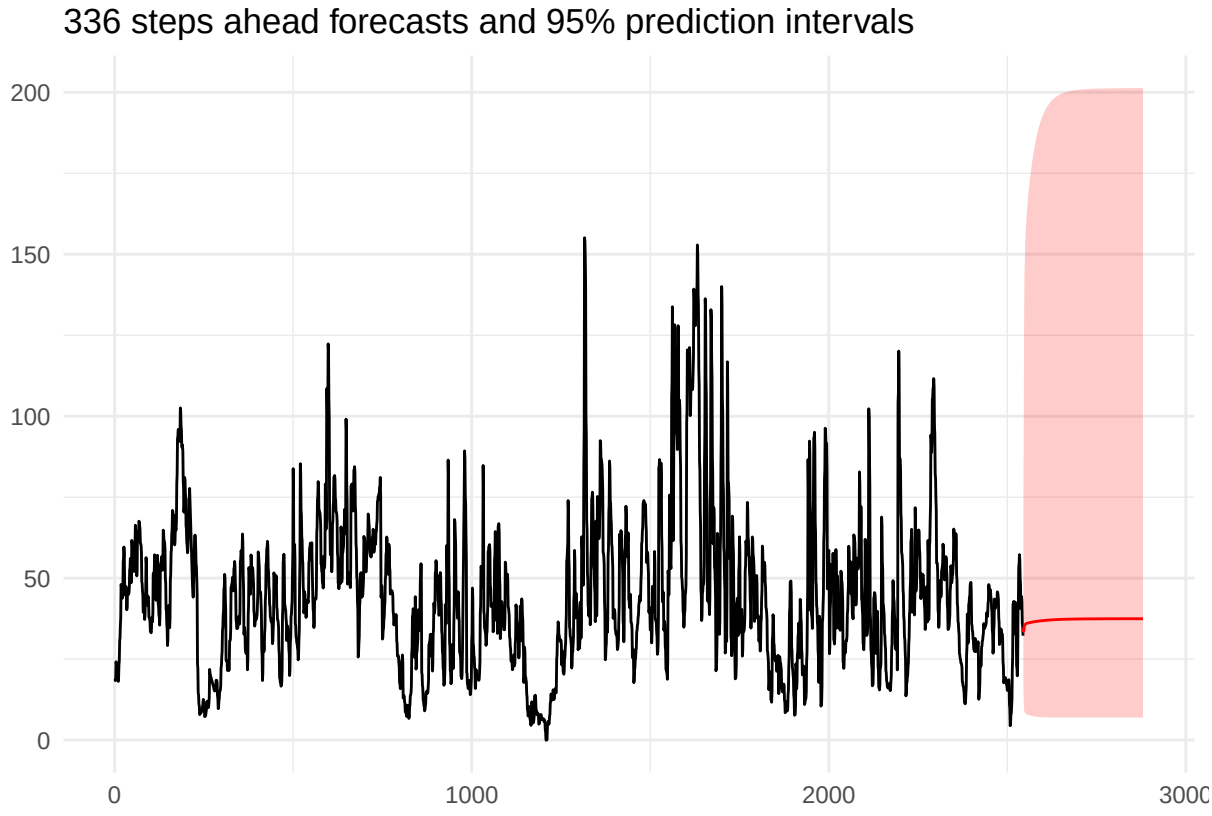

residuals
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0
5
15
25
35
Lag


Normal Q−Q Plot of Std Residuals
Sample Quantiles
80
60
40
20
0
−20
−60
−3
−1
0
1
2
3
Theoretical Quantiles

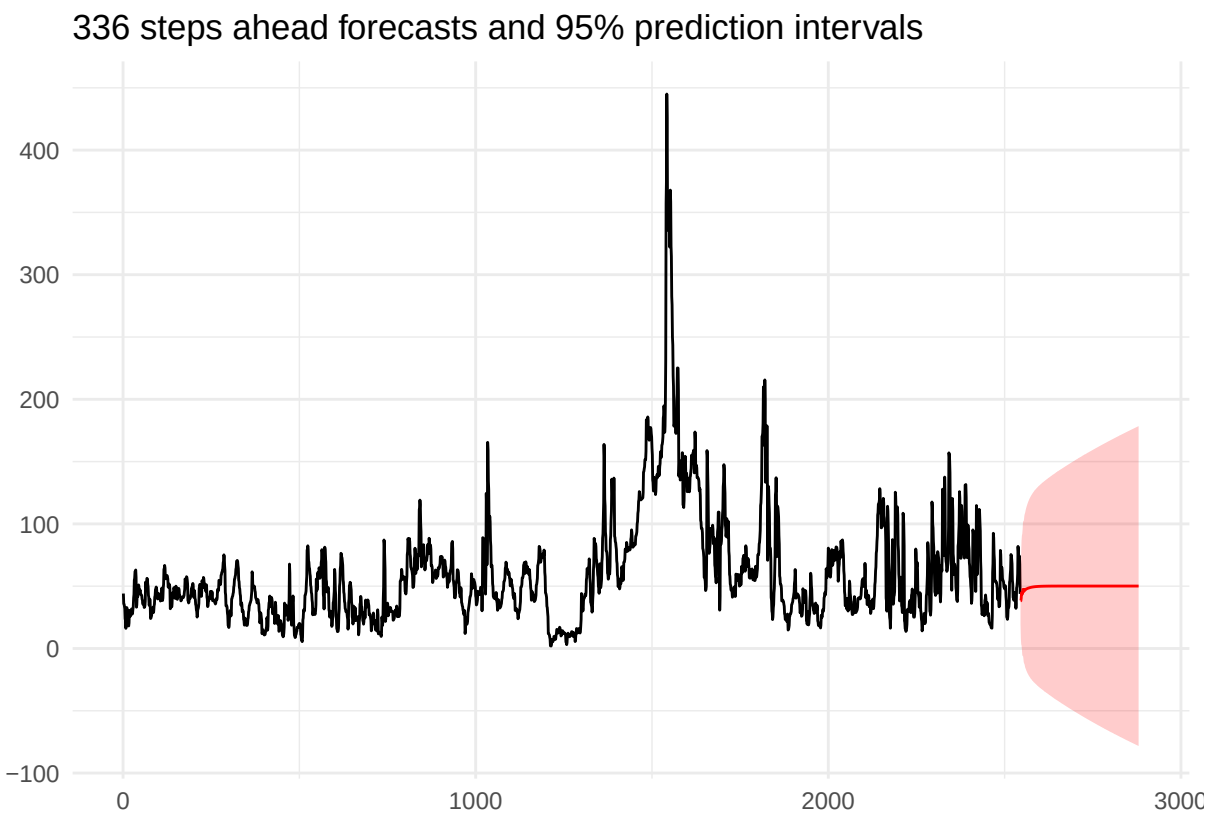


4.2 Forecasts and 95% prediction intervals for each city

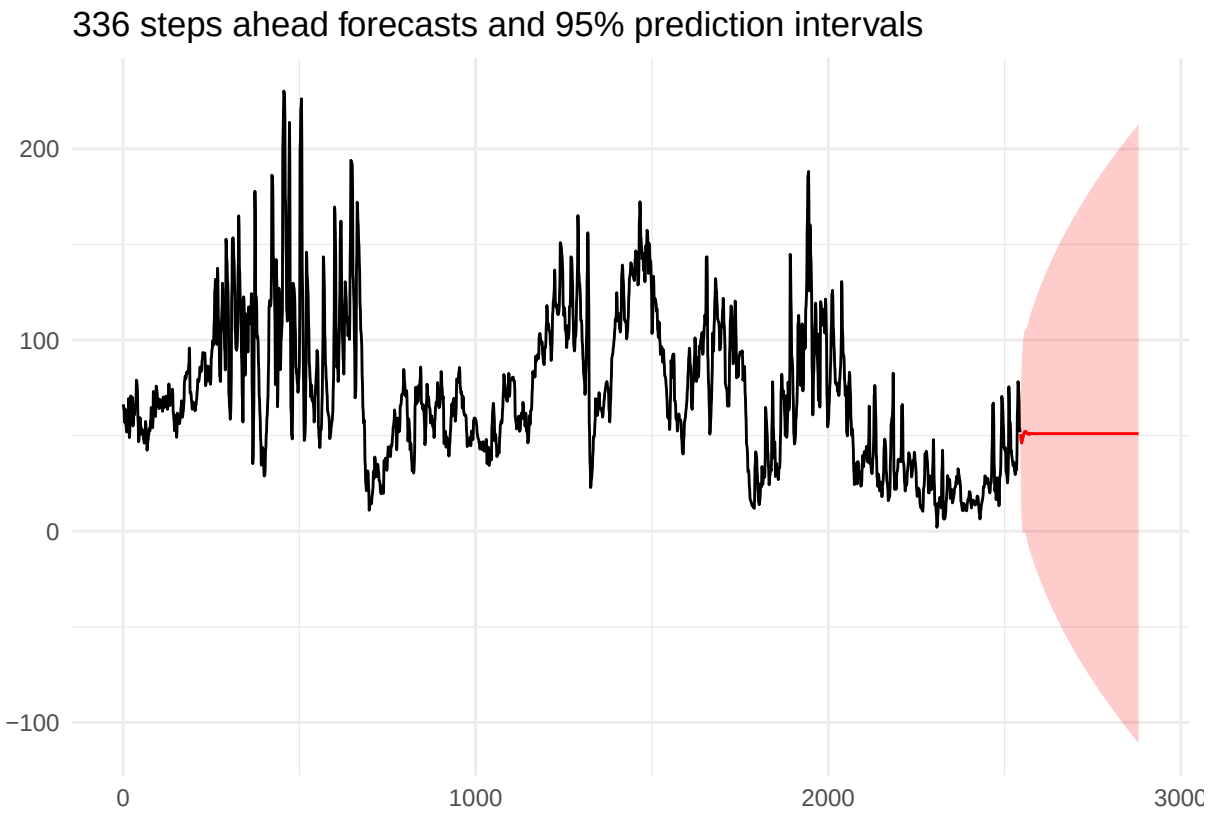
4.2.1 Forecasts and 95% prediction intervals for city1



4.2.2 Forecasts and 95% prediction intervals for city2



4.2.3 Forecasts and 95% prediction intervals for city3



5 Appendix

5.1 Expanding Window cross-validation for hydro data

```
EWCV.logSARIMA <- function(data, p, d, q, P, D, Q, S) {
  len <- nrow(data)
  data.train.percent <- c(0.5, 0.6, 0.7, 0.8, 0.9)
  i <- 1
  MSE <- c()
  for (percent in data.train.percent) {
    num.train <- round(len * percent)
    num.test <- num.train + 1
    data.train <- data$Value[1:num.train]
    data.test <- data$Value[num.test:len]
    len.test <- length(data.test)

    model <- astsa::sarima(data.train, p, d, q, P, D, Q, S)
    pred <- as.numeric(sarima.for(as.ts(log(data.train)), len.test,
                                  p, d, q, P, D, Q, S, plot = FALSE)$pred)
    MSE[i] <- mean((data.test - exp(pred))^2)
    i <- i + 1
  }
  return(mean(MSE))
}
```

5.2 Expanding Window cross-validation for stock data on standard GARCH

```
EWCV.GARCH <- function(data, formula) {
  len <- nrow(data)
  data.train.percent <- c(0.5, 0.6, 0.7, 0.8, 0.9)
  i <- 1
  MSE <- c()
  for (percent in data.train.percent) {
    num.train <- round(len * percent)
    num.test <- num.train + 1
    data.train <- data$Value[1:num.train]
    data.test <- data$Value[num.test:len]
    len.test <- length(data.test)
    model <- garchFit(formula = formula, data = data.train, trace = FALSE)
    pred <- predict(model, len.test)$meanForecast[1]
    MSE[i] <- mean((data.test - pred)^2)
    i <- i + 1
  }
  return(mean(MSE))
}
```

5.3 Expanding Window cross-validation for stock data on exponential GARCH

```
EWCV.eGARCH <- function(data, spec) {
  len <- nrow(data)
  data.train.percent <- c(0.5, 0.6, 0.7, 0.8, 0.9)
  i <- 1
  MSE <- c()
```

```
  for (percent in data.train.percent) {
    num.train <- round(len * percent)
    num.test <- num.train + 1
    data.train <- data$Value[1:num.train]
    data.test <- data$Value[num.test:len]
    len.test <- length(data.test)
    model <- rugarch::ugarchfit(spec, data = data.train)
    pred <- as.numeric(fitted(rugarch::ugarchforecast(model, n.ahead = len.test)))
    MSE[i] <- mean((data.test - pred)^2)
    i <- i + 1
  }
  return(mean(MSE))
}
```