Project 2

2022-12-04

**Library Setup:**

Here, we are importing the necessary libraries.

require("knitr")

## Loading required package: knitr

sourcedir <-"C:/Users/Neha/Documents/SYS4021/RCode"

opts\_knit$set(root.dir = sourcedir) library(forecast)

## Warning: package ’forecast’ was built under R version 4.1.3

## Registered S3 method overwritten by ’quantmod’:

## method from

## as.zoo.data.frame zoo

library(mtsdi)

## Warning: package ’mtsdi’ was built under R version 4.1.3

## Loading required package: gam

## Warning: package ’gam’ was built under R version 4.1.3

## Loading required package: splines

## Loading required package: foreach

## Warning: package ’foreach’ was built under R version 4.1.3

## Loaded gam 1.22

## mtsdi 0.3.5

library(MTS)

## Warning: package ’MTS’ was built under R version 4.1.3

library(ggplot2) library(ggpubr)

## Warning: package ’ggpubr’ was built under R version 4.1.2

##

## Attaching package: ’ggpubr’

## The following object is masked from ’package:forecast’:

##

## gghistogram

library(ggfortify)

## Warning: package ’ggfortify’ was built under R version 4.1.3

## Registered S3 methods overwritten by ’ggfortify’:

## method from

## autoplot.Arima forecast ## autoplot.acf forecast ## autoplot.ar forecast ## autoplot.bats forecast

## autoplot.decomposed.ts forecast

## autoplot.ets forecast ## autoplot.forecast forecast ## autoplot.stl forecast ## autoplot.ts forecast ## fitted.ar forecast ## fortify.ts forecast ## residuals.ar forecast

library(tseries)

## Warning: package ’tseries’ was built under R version 4.1.3

library(lubridate)

##

## Attaching package: ’lubridate’

## The following objects are masked from ’package:base’:

##

## date, intersect, setdiff, union

# Imputing Data

|  |
| --- |
| setwd(sourcedir)  airquality = read.csv('AirQualityUCI.csv')  *# replace -200 with NA* airquality[airquality == -200] <- NA  *# convert integer type to numeric* intcols = c(4,5,7,8,9,10,11,12) **for**(i **in** 1:length(intcols)){ airquality[,intcols[i]] <- as.numeric(airquality[,intcols[i]])  }  setwd(sourcedir)  *# create new data frame with just CO and NO2*  AQdata = airquality[,c(3,10)]  *# impute missing air quality data* f <- **~** CO.GT. + NO2.GT.  t <- c(seq(1,dim(AQdata)[1],1))  i <- mnimput(f, AQdata, eps=1e-3, ts=TRUE, method='gam',  ga.control=list(formula=paste(names(AQdata)  [c(1:2)],'~ns(t,2)')))  *# set airquality to imputed data*  AQdata <- i$filled.dataset  *# aggregate to daily maxima for model building* dailyAQ <- aggregate(AQdata, by=list(as.Date(airquality[,1],"%m/%d/%Y")), FUN=max) |

# Question 1

We start by extracting the NO2 values from dailyAQ and converting them to a time series. The plot of the time series values can be seen in Figure 1.

|  |
| --- |
| *#making this a time series* no2.ts <-ts(dailyAQ$NO2.GT.)  *#Figure 1:* plot(no2.ts) |

no2.ts

0

100

200

300

400

100

150

200

250

300

350

Time

fig.dim = c(8, 6)

**(a) Here, we’re discovering and modeling any seasonal components. To test for seasonality, we can use the acf command. There seems to be seasonality since the graph exhibits sinusoidal decay. Based on the ACF plot (Figure 2) of the NO2 time series, it can be observed that the ACF is a decreasing sinusoidal with a peak occurring roughly every 7 day, which indicates a weekly seasonality. In the periodogram (Figure 3), the observed period of the first, largest peak is 192 days, however this data may be attributed to red noise as the second largest peak has a period of 7 days and better supports the results of the ACF.**

*#acf chart -> displays sinusoidal decay, peaks every 7 days indicating weekly seasonality*

*#Figure 2:*

no2.acf <- acf(no2.ts, plot=FALSE)

plot(no2.acf, main = "Figure 2: ACF of NO2 Time Series")

## Figure 2: ACF of NO2 Time Series

0

5

10

15

20

25

0.0

0.2

0.4

0.6

0.8

1.0

ACF

Lag

*#Periodogram for seasonality*

pg.no2 <- spec.pgram(no2.ts,spans=9,demean=T,log='no',plot=F)

spec.no2 <- data.frame(freq=pg.no2$freq, spec=pg.no2$spec) ggplot(spec.no2) + geom\_line(aes(x=freq,y=spec), width=6, height=6) +

ggtitle(" Figure 3: Smooth Periodogram of No2")

## Warning: Ignoring unknown parameters: width, height

### Figure 3: Smooth Periodogram of No2

0

5000

10000

15000

20000

25000

0.0

0.1

0.2

0.3

0.4

0.5

freq

spec

|  |
| --- |
| *#Finding the peak* max.omega.no2<-spec.no2$freq[which(pg.no2$spec==max(pg.no2$spec))]  *#Frequency of the peak* max.omega.no2 |

## [1] 0.005

*#Period of the peak*

1/max.omega.no2

## [1] 200

|  |
| --- |
| *#Finding the periods of the next biggest peaks*  sorted.spec <- sort(pg.no2$spec, decreasing=T, index.return=T)  sorted.omegas <- pg.no2$freq[sorted.spec$ix] sorted.Ts <- 1/pg.no2$freq[sorted.spec$ix] sorted.omegas[1:20] *#peak of ~7.24 days around second peak* |

## [1] 0.0050 0.0075 0.0025 0.0100 0.0125 0.0150 0.0175 0.0200 0.0225 0.0250 ## [11] 0.1500 0.1475 0.1450 0.0300 0.1425 0.1400 0.0275 0.0425 0.1375 0.0400 *#(frequency of around 0.15)* sorted.Ts[1:20]

## [1] 200.000000 133.333333 400.000000 100.000000 80.000000 66.666667 ## [7] 57.142857 50.000000 44.444444 40.000000 6.666667 6.779661

## [13] 6.896552 33.333333 7.017544 7.142857 36.363636 23.529412 ## [19] 7.272727 25.000000

We created a seasonal trend model which predicts the no2 time series based on time as well as accounting for sinusoidal seasonality. The model is statistically significant with a p-value of less than 2.2e-16. The coefficient for the time variable is 0.24817 which means that for every 1 increase in time, there should be about a 0.24817 increase in NO2 levels. The sine term is statistically significant indicating that there is a sinusoidal seasonal relationship between time and NO2 levels. Since we found that there was weekly seasonality, we decided to set the period to every 7 days.

Based on the graph of seasonal trend (Figure 4), we can see that there are seasons or a sinusoidal relationship between time and NO2 levels. We also ran diagnostics for the seasonal trend model (Figure 5) and saw that there was good normality in the QQ plot with a minor Gaussian tail. The Residuals vs Fitted plot shows that the residuals are somewhat scattered around the zero line with significant trends at certain peaks.

|  |
| --- |
| *#creating the seasonality model* time.no2<-c(1:(length(no2.ts)))  no2.trend.seasonal <- lm(no2.ts ~ time.no2 + sin(2\*pi\*time.no2/7) + cos(2\*pi\*time.no2/7))  summary(no2.trend.seasonal) |

##

## Call:

## lm(formula = no2.ts ~ time.no2 + sin(2 \* pi \* time.no2/7) + cos(2 \*

## pi \* time.no2/7))

##

## Residuals:

## Min 1Q Median 3Q Max ## -101.153 -28.634 0.959 26.687 138.096

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 116.17036 4.19832 27.671 < 2e-16 \*\*\* ## time.no2 0.24817 0.01856 13.370 < 2e-16 \*\*\* ## sin(2 \* pi \* time.no2/7) 15.36453 2.95925 5.192 3.37e-07 \*\*\* ## cos(2 \* pi \* time.no2/7) 5.61983 2.96659 1.894 0.0589 .

## ---

## Signif. codes: 0 ’\*\*\*’ 0.001 ’\*\*’ 0.01 ’\*’ 0.05 ’.’ 0.1 ’ ’ 1

##

## Residual standard error: 41.43 on 387 degrees of freedom

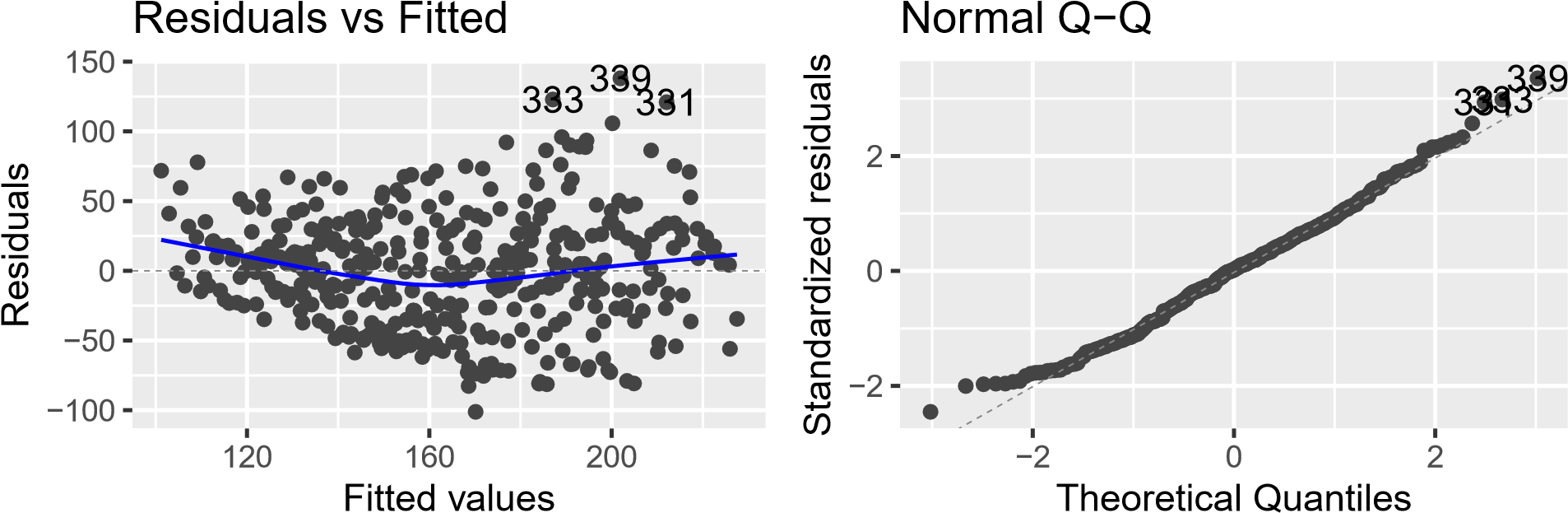
## Multiple R-squared: 0.3491, Adjusted R-squared: 0.344 ## F-statistic: 69.17 on 3 and 387 DF, p-value: < 2.2e-16

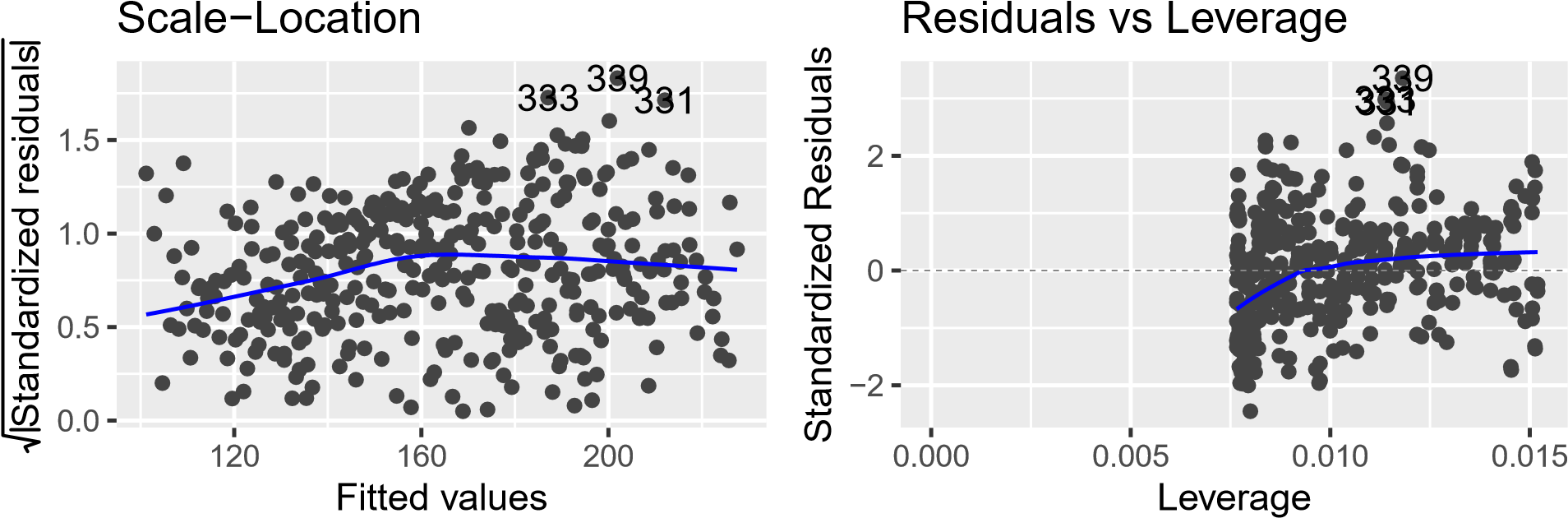
|  |
| --- |
| *#Figure 4:*  graph.seasonal.no2 <- ggplot(dailyAQ, aes(x=Group.1,y=NO2.GT.)) + geom\_line() + geom\_line(aes(x=Group.1,y=no2.trend.seasonal$fitted.values),color="red", width=6, height=6) +  xlab("") + ylab("Figure 4: NO2 Seasonality Graph") |

## Warning: Ignoring unknown parameters: width, height

*#Figure 5:*

autoplot(no2.trend.seasonal, labels.id = NULL,width=6, height=6)





**(b) Here, we are discovering and modeling any trends. First, we started by creating a time variable using only data points in no2. Then, we built a model that predicts the time series of no2 based on the newly created time variable. Based on the model diagnostics, the model is significant with a p-value of less than 2.2e-16. The coefficient for the time variable is 0.25646, which means that for every 1 increase in unit of time, there is a 0.25646 increase in NO2 levels. The time term is also statistically significant indicating that there is a relationship between duration and NO2 levels. Based on the graph (Figure 6), the trend line shows that there is a positive linear relationship between time and NO2 levels. The graph also seems to show that time series is has additive seasonal changes.**

We also ran diagnostics for the plot (Figure 7). The QQ plot shows a good fit of normality with no Gaussian tails. The Residuals vs Fitted plot shows that the residuals are somewhat scattered around the zero line which indicates that the assumption that the relationship is somewhat linear is reasonable. The residuals vs fitted plot also shows several outliers, particularly obvervations 339,331,and 333. The scale-location plot indicates a small lack of fit as shown by the non-horizontal trend line. This could mean that there are other predictors that could explain the variance in the dataset, or that the outliers are skewing the data.

*#New model called no2.trend which predicts no2.ts based on the time variable* no2.trend <- lm(no2.ts[time.no2] ~ time.no2) summary(no2.trend)

##

## Call:

## lm(formula = no2.ts[time.no2] ~ time.no2)

##

## Residuals:

## Min 1Q Median 3Q Max

## -85.674 -34.168 1.771 28.437 139.893

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 116.40204 4.34936 26.76 <2e-16 \*\*\* ## time.no2 0.24692 0.01923 12.84 <2e-16 \*\*\*

## ---

## Signif. codes: 0 ’\*\*\*’ 0.001 ’\*\*’ 0.01 ’\*’ 0.05 ’.’ 0.1 ’ ’ 1

##

## Residual standard error: 42.92 on 389 degrees of freedom ## Multiple R-squared: 0.2977, Adjusted R-squared: 0.2959 ## F-statistic: 164.9 on 1 and 389 DF, p-value: < 2.2e-16

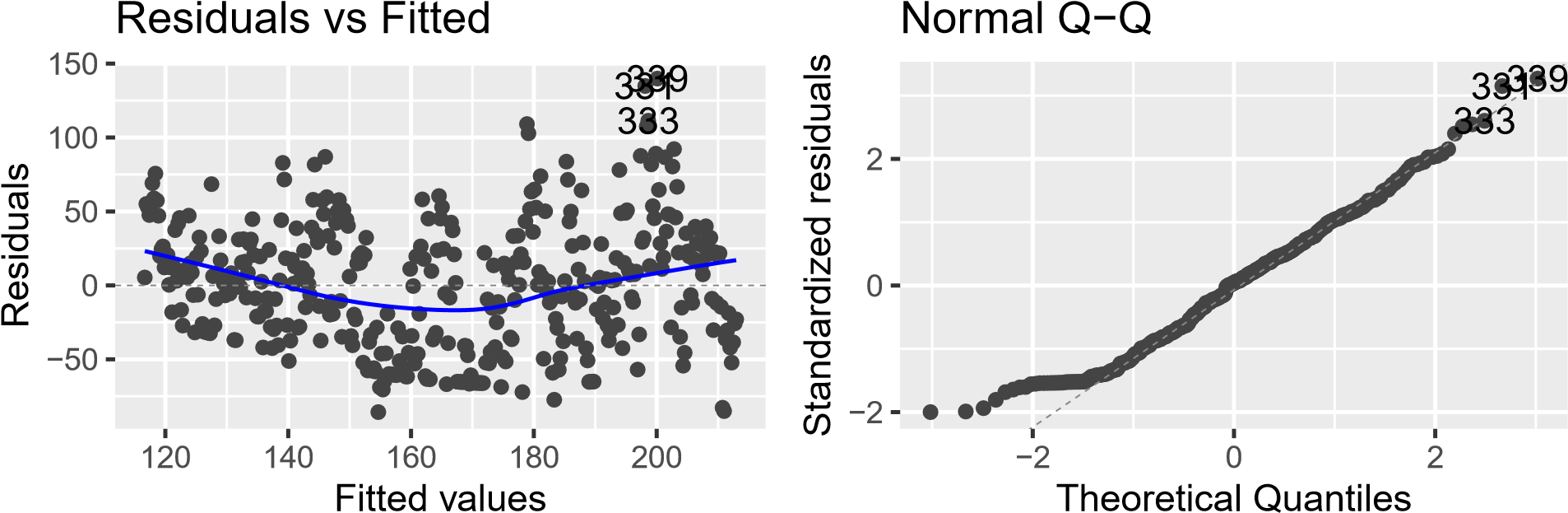
|  |
| --- |
| *#Figure 6:*  graph.trend.no2 <- ggplot(dailyAQ, aes(x=Group.1,y=NO2.GT.)) + geom\_line() + stat\_smooth(  ylab("Figure 6: Trend Graph of NO2") |

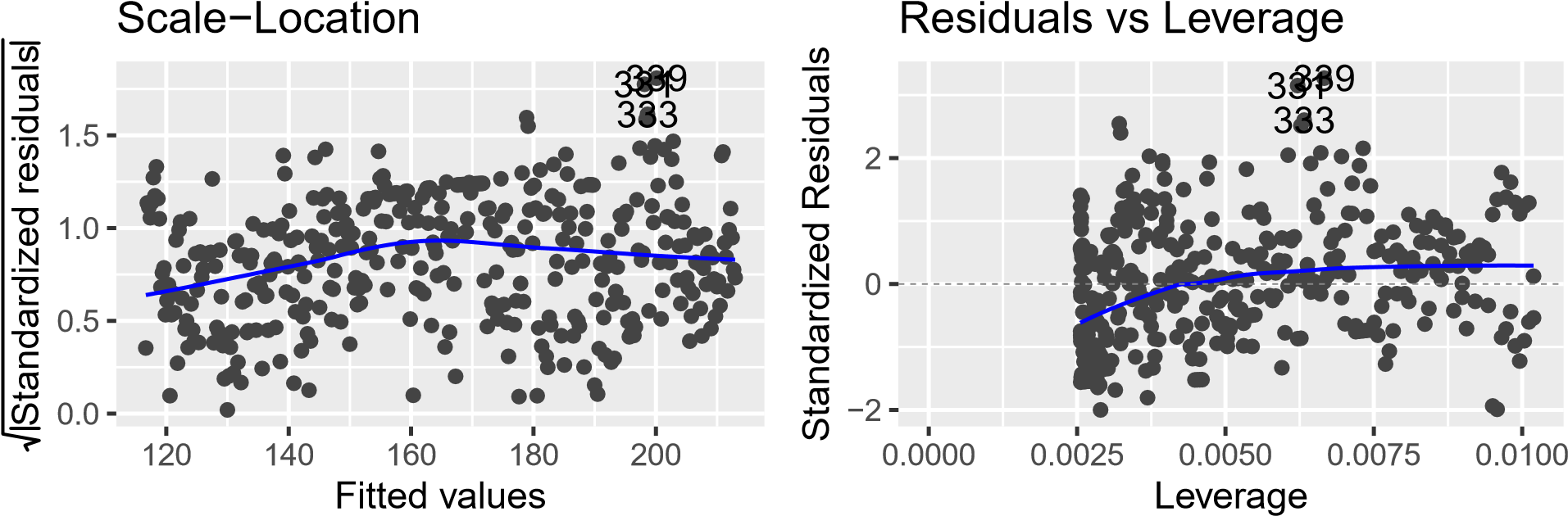
method="lm",col

## Warning: Ignoring unknown parameters: width, height

*#Figure 7:*

autoplot(no2.trend, labels.id = NULL,width=6, height=6)





In addition to the trend model, we also built a trendseason model which models trend while accounting for weekly seasonality. In order to do this, we built dummy variables modeling each day of the week in relation to the dates in our dataset. We then built a linear regression model using both time and the dummy variables as explanatory variables and the time series as the response variable. We found that the model is significant with a p-value of 2.2e-16. The time variable is also significant with a coefficient of 0.25749 indicating that for every 1 increase in time, there is a 0.25749 increase in NO2 levels. In terms of the dummy variables, we found that only Saturday and Sunday are statistically significant with coefficients of -26.67938 and -39.97451, respectively. This indicates that the likelihood of NO2 levels being impacted by the day of week, specifically on Saturdays and Sundays, may not be due to random chance. Figure 8 shows the trend+seasonality of the time series data.

We also ran diagnostics for the plot (Figure 9). The QQ plot shows a good fit of normality with a very minor Gaussian tail. The Residuals vs Fitted plot shows that the residuals are somewhat scattered around the zero line which indicates that the assumption that the relationship is somewhat linear is reasonable. The residuals vs fitted plot also shows several outliers, particularly observations 339,331,and 333. The scale-location plot indicates a small lack of fit as shown by the non-horizontal trend line. This could mean that there are other predictors that could explain the variance in the dataset, or that the outliers are skewing the data.

|  |
| --- |
| *#Model the seasonality for no2 data set using dummy variables. Use day of the week as the interval.*  no2.day <- time.no2 %% 7 no2.day <-as.factor(time.no2 %% 7) fulltime = c(1:length(no2.ts)) Day <- rep(NA, length(no2.ts)) |

Day[which((fulltime %% 7) == 1)] <- "W"

Day[which((fulltime %% 7) == 2)] <- "Th"

Day[which((fulltime %% 7) == 3)] <- "Fri" Day[which((fulltime %% 7) == 4)] <- "Sat"

Day[which((fulltime %% 7) == 5)] <- "Sun"

Day[which((fulltime %% 7) == 6)] <- "M"

Day[which((fulltime %% 7) == 0)] <- "T"

Day <- as.factor(Day)

*#building a trendseason model* no2.trendseason<-lm(no2.ts[time.no2]~time.no2+Day[time.no2])

*#Is no2.trendseason significant?*

summary(no2.trendseason)

##

## Call:

## lm(formula = no2.ts[time.no2] ~ time.no2 + Day[time.no2])

##

## Residuals:

## Min 1Q Median 3Q Max

## -93.791 -27.894 1.166 24.649 128.062

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 127.90546 6.53950 19.559 < 2e-16 \*\*\* ## time.no2 0.24788 0.01834 13.517 < 2e-16 \*\*\*

## Day[time.no2]M -9.39653 7.73463 -1.215 0.225166

## Day[time.no2]Sat -25.94796 7.73446 -3.355 0.000873 \*\*\*

## Day[time.no2]Sun -38.76853 7.73452 -5.012 8.23e-07 \*\*\*

## Day[time.no2]T -4.20795 7.76952 -0.542 0.588411 ## Day[time.no2]Th 0.14057 7.73446 0.018 0.985509 ## Day[time.no2]W -3.53620 7.73452 -0.457 0.647789

## ---

## Signif. codes: 0 ’\*\*\*’ 0.001 ’\*\*’ 0.01 ’\*’ 0.05 ’.’ 0.1 ’ ’ 1

##

## Residual standard error: 40.93 on 383 degrees of freedom ## Multiple R-squared: 0.3712, Adjusted R-squared: 0.3597 ## F-statistic: 32.3 on 7 and 383 DF, p-value: < 2.2e-16

*# Figure 8:*

plot(no2.ts, main = "Figure 8: Plot of Trend+Season Model for NO2 Values")

lines(ts(no2.trendseason$fitted.values),col="red")

## Figure 8: Plot of Trend+Season Model for NO2 Values

no2.ts

0

100

200

300

400

100

150

200

250

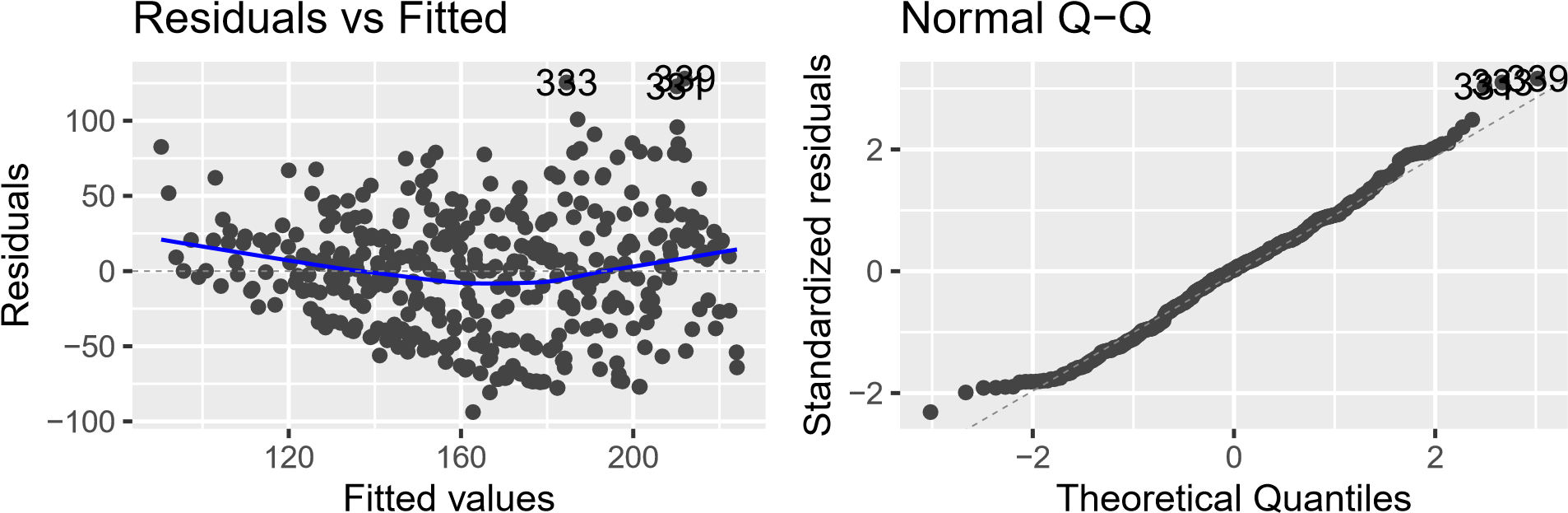
300

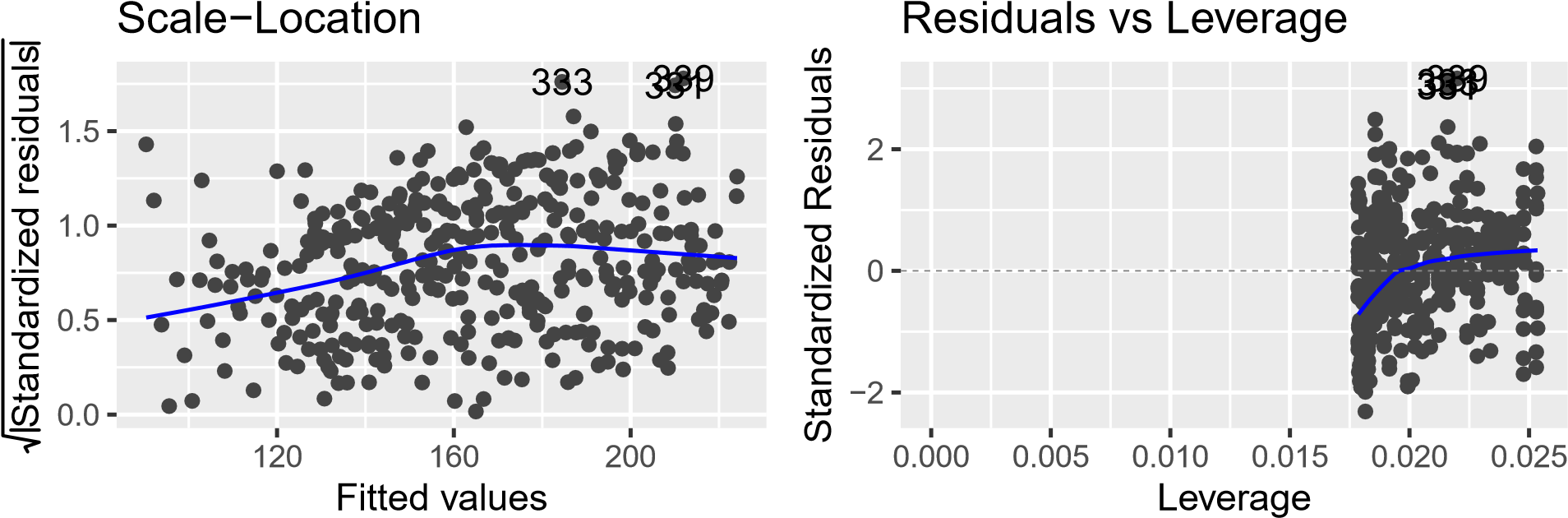
350

Time

*#Figure 9:*

autoplot(no2.trendseason, labels.id = NULL,width=6, height=6)





**(c) Here, we are looking at autoregressive or moving average components. A weekly stationary time series is one that has a constant mean with autocorrelation as a function of the lag. Because the acf does not have a slow or linear decay, it is clear that it is a stationary time series. Based on the ACF and PACF plot of the residuals from no2.trendseason (Figure 10), the data displays autoregressive components as the ACF shows a sinusoidal decrease. Additionally, the PACF only displays a small number, 2, of lags before it cuts off.**

*# Getting the residuals from no2.trendseason model above and store in e.ts.no2:*

e.ts.no2 <- ts(no2.trendseason$residuals)

*#Figure 10:*

autoplot(e.ts.no2,width=6, height=6)

−100

−50

0

50

100

0

100

200

300

400

|  |
| --- |
| *#the ACF for the residuals* no2.acf <- ggAcf(e.ts.no2)  *#PACF for the residuals* no2.pacf <- ggPacf(e.ts.no2)  *# Plot acf and pacf side by side for easier examination* plot <- ggarrange(no2.acf,no2.pacf,nrow=2,ncol=1)  annotate\_figure(plot, top = text\_grob("Figure 10: ACF and PACF Side by Side")) |

Figure 10: ACF and PACF Side by Side

### Series: e.ts.no2

0.0

0.2

0.4

0.6

0

5

10

15

20

25

Lag

ACF

### Series: e.ts.no2

0.0

0.2

0.4

0.6

0

5

10

15

20

25

Lag

PACF

**(d) We built four different types of models: autoregressive, moving average, autoregressive moving average, and autoregressive integrated moving average. For each model, we decided build two versions: one with the mean and one without the mean. We looked at the AIC values for each and chose the model that had the lower value. After we found our best model for the different types, we decided to compare all four of the chosen ones using AIC, BIC, and the Ljung-Box test. We found that the auto model has the best AIC, the autoregressive model (AR) has the best BIC, and the moving average (MA) and the autoregressive moving average (ARMA) have the best Ljung-Box statistic with p-values above**

**0.05.**

Looking at AIC value is useful for finding the best model for predicting future observations. Based on that, we choose to use an autoregressive integrated moving average model (as defined by our variable no2.auto) going forward. The model has p,d,q values of 3,1,1 respectively. Looking at the residuals vs fitted plot (Figure 11) for our overaggressive integrated moving average model, we can see that the residuals bounce randomly around the zero line and also no one residual stands out. These are characteristics of a well behaved residuals vs fitted plot. The QQ plot for our autoregressive integrated moving average model seems to display decent normality with a slight upper Gaussian tail. Looking at Ljung-Box test (Figure 12), we can see that all of the points have a p-value of above 0.05 which indicates that the residuals are independent. Based on everything above, our autoregressive integrated moving average model is a good fit for the data. Further improvements in the diagnostics can be made by addressing the slight Gaussian tail in the QQ plot.

|  |
| --- |
| *#AR Model #ar(1) p=2*  no2.ar1 <- arima(e.ts.no2, order=c(2,0,0))  no2.ar1 <- arima(e.ts.no2, order=c(3,0,0), include.mean=FALSE)  *#better model compared to first one*  *#MA Model: #ma(12) p=0, q=14*  no2.ma12 <- arima(e.ts.no2, order=c(0,0,14))  no2.ma12 <- arima(e.ts.no2, order=c(0,0,14), include.mean=FALSE)  *#better model compared to first one*  *#ARMA Model*  *#arma(1,12) p=2, q=14*  no2.arma12 <- arima(e.ts.no2, order=c(2,0,14))  no2.arma12 <- arima(e.ts.no2, order=c(2,0,14), include.mean=FALSE) *#better model compared to first*  *#ARIMA AUTO:*  no2.auto <- auto.arima(e.ts.no2) no2.auto <- auto.arima(e.ts.no2,approximation=FALSE) *#better arima*  *#AIC comparison*  AIC(no2.ar1) |

## [1] 3775.017

AIC(no2.ma12)

## [1] 3786.159

AIC(no2.arma12)

## [1] 3790.275

AIC(no2.auto) *#best model*

## [1] 3773.473

*#BIC comparison*

BIC(no2.ar1) *#best model*

## [1] 3790.892 BIC(no2.ma12)

## [1] 3845.69

BIC(no2.arma12)

## [1] 3857.743

BIC(no2.auto)

## [1] 3785.379

|  |
| --- |
| *# assess residuals vs. fitted*  model1 = ggplot() + geom\_point(aes(x=fitted(no2.ar1), y=no2.ar1$residuals,width=6, height=6)) +  ggtitle("AR1") |

## Warning: Ignoring unknown aesthetics: width, height

|  |
| --- |
| model2 = ggplot() + geom\_point(aes(x=fitted(no2.ma12), y=no2.ma12$residuals,width=6, height=6)) +  ggtitle("MA2") |

## Warning: Ignoring unknown aesthetics: width, height

|  |
| --- |
| model3 = ggplot() + geom\_point(aes(x=fitted(no2.arma12), y=no2.arma12$residuals,width=6, height=6)) +  ggtitle("ARMA12") |

## Warning: Ignoring unknown aesthetics: width, height

|  |
| --- |
| model4 = ggplot() + geom\_point(aes(x=fitted(no2.auto), y=no2.auto$residuals,width=6, height=6)) +  ggtitle("Auto") |

## Warning: Ignoring unknown aesthetics: width, height

best\_model\_plot <- ggarrange(model1, model2, model3, model4, ncol=2, nrow=2)

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

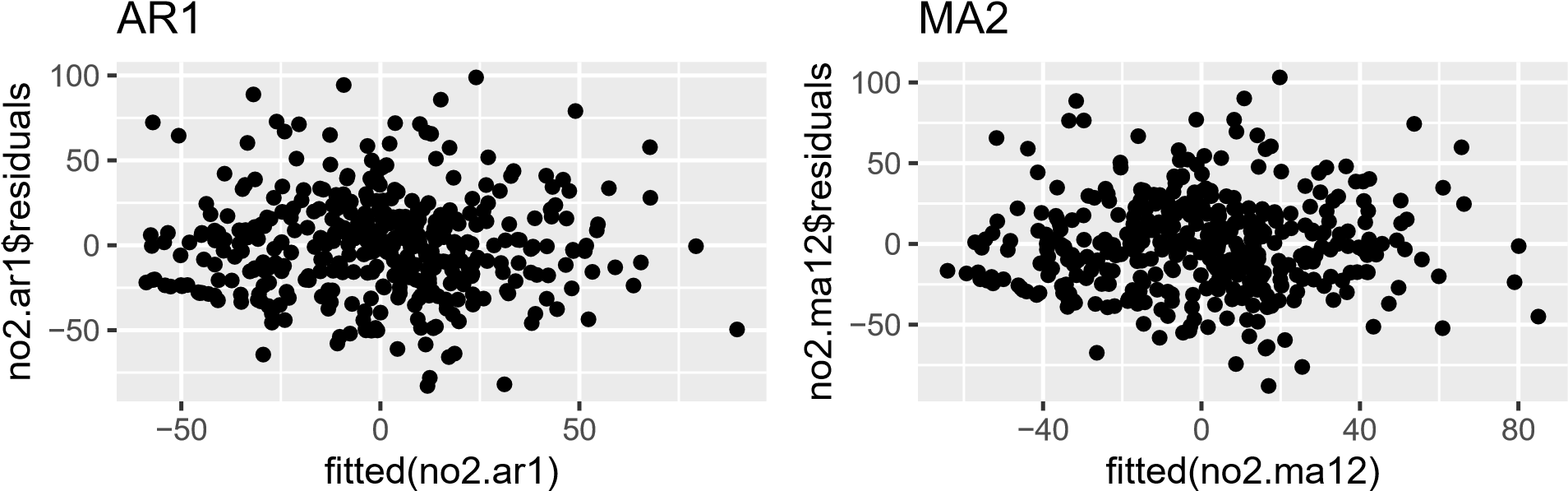
## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

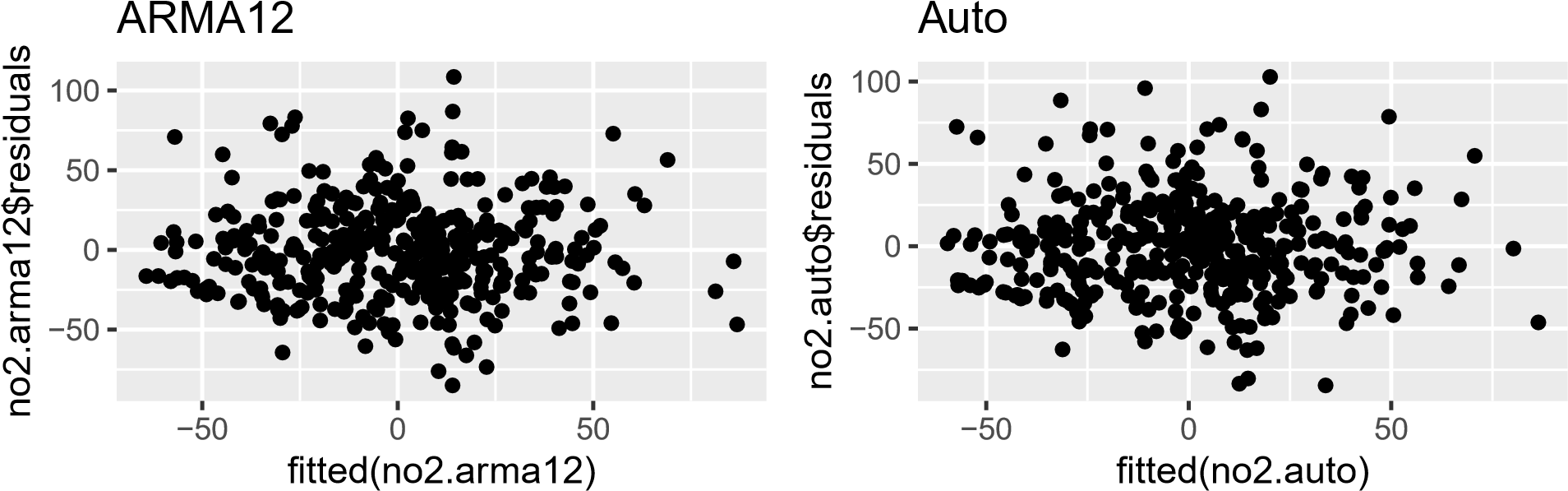
## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous. annotate\_figure(best\_model\_plot, top = text\_grob("Figure 11: Residuals of Chosen Models"))

Figure 11: Residuals of Chosen Models





*#QQ Plots*

ggtsdiag(no2.ar1,gof.lag=20)

2

0

0 100 200 300 400

### ACF of Residuals

0.00

0.25

0.50

0.75

1.00

0

5

10

15

20

25

Lag

ACF

### p values for Ljung−Box statistic

0.00

0.25

0.50

0.75

1.00

5

10

15

20

Lag

p value

ggtsdiag(no2.ma12,gof.lag=20) *#better box test*

2

0

0 100 200 300 400

### ACF of Residuals

0.00

0.25

0.50

0.75

1.00

0

5

10

15

20

25

Lag

ACF

### p values for Ljung−Box statistic

0.00

0.25

0.50

0.75

1.00

5

10

15

20

Lag

p value

ggtsdiag(no2.arma12,gof.lag=20) *#better box test*

0

2

4

0

100

200

300

400

### ACF of Residuals

0.00

0.25

0.50

0.75

1.00

0

5

10

15

20

25

Lag

ACF

p values for Ljung−Box statistic

0.00

0.25

0.50

0.75

1.00

5

10

15

20

Lag

p value

ggtsdiag(no2.auto,gof.lag=20)

2

0

0 100 200 300 400

### ACF of Residuals

0.00

0.25

0.50

0.75

1.00

ACF

0 5 10 15 20 25

Lag

### p values for Ljung−Box statistic

0.00

0.25

0.50

0.75

1.00

p value

5 10 15 20

Lag

|  |
| --- |
| *# assess normality of residuals*  model1 = qplot(sample=no2.ar1$residuals) + stat\_qq\_line(color="red") + ggtitle("AR1")  model2 = qplot(sample=no2.ma12$residuals) + stat\_qq\_line(color="red") + ggtitle("MA2")  model3 = qplot(sample=no2.arma12$residuals) + stat\_qq\_line(color="red") + ggtitle("ARMA12")  model4 = qplot(sample=no2.auto$residuals) + stat\_qq\_line(color="red") + ggtitle("Auto") plot2 <- ggarrange(model1, model2, model3, model4, ncol=2, nrow=2) |

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

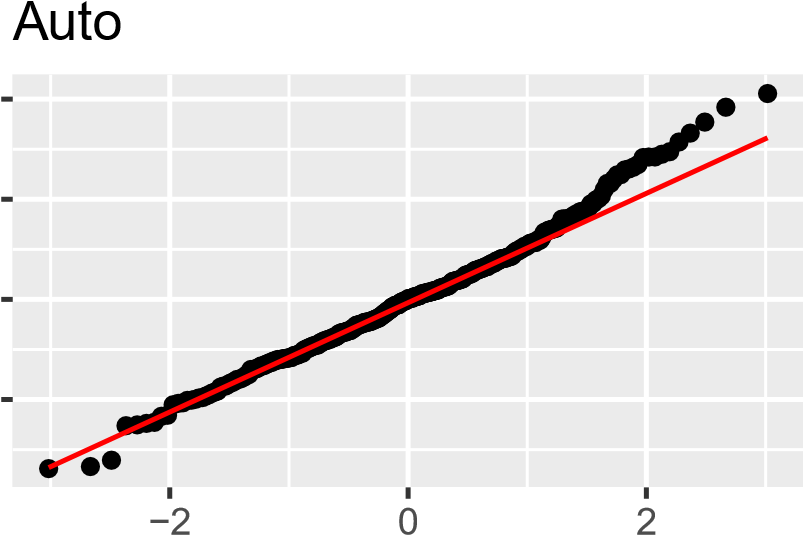
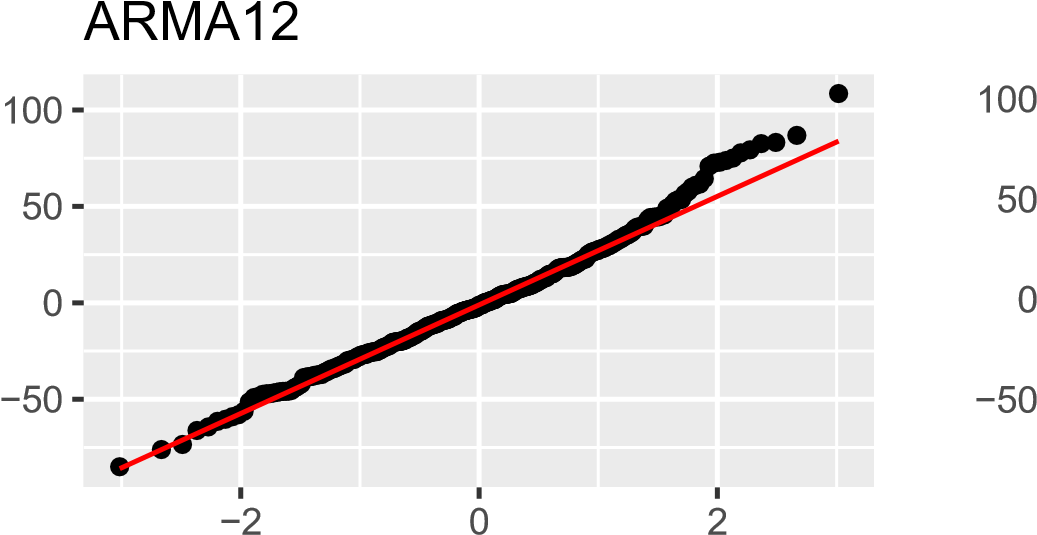
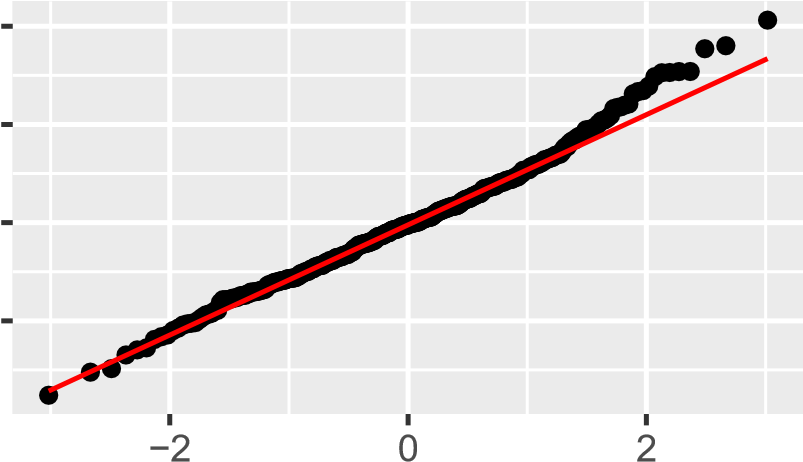
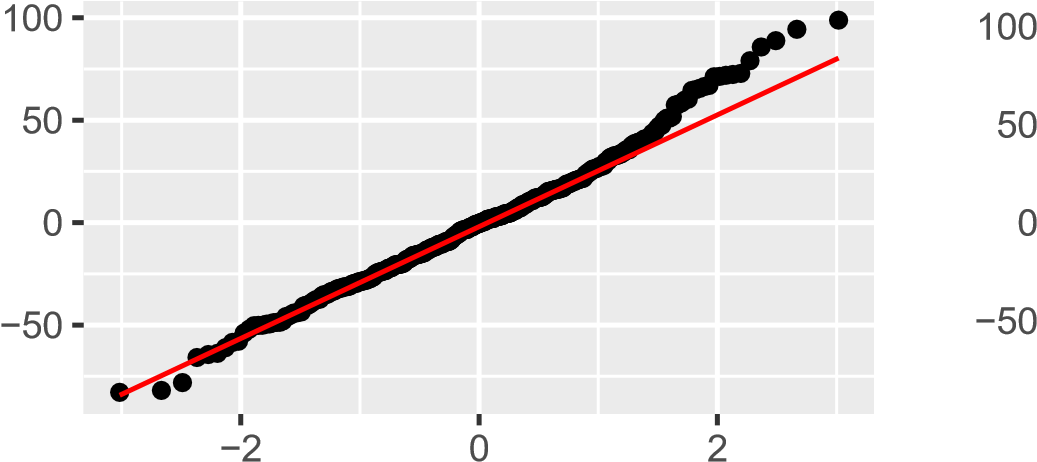
## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous. ## Don’t know how to automatically pick scale for object of type ts. Defaulting to continuous.

annotate\_figure(plot2, top = text\_grob("Figure 12: Normality of Chosen Models"))

Figure 12: Normality of Chosen Models

### AR1 MA2



## (e) Here, we forecasted the next 7 days of NO2 concentrations using our chosen autoregressive integrated moving average model. We created a new time variable for the next 7 days of the dataset. We then made a dataframe which sets time to be the next 7 days and only looks at the last 7 values of our NO2 time series dataset. We then predicted the next 7 days using our trendseason model and forecasted using our autoregressive integrated moving average model.

Our mean square error for the model was 36023.04 which is relatively high indicating that our model was not very accurate at predicting the last 7 days of NO2 observations. Moreover, plotting the predicted and actual values (Figure 13), we can see a significant gap in the values further reinforcing our point.

|  |
| --- |
| *#a next 7 day time variable and dataframe*  next.7day.time <- c((length(time.no2)-6):(length(time.no2))) next.7day <- data.frame(time.no2 = next.7day.time, no2.ts[next.7day.time]) *#7day for no2 full*  *# Prediction for the next 7days by no2.trendseason and forecast by no2.auto* no2.predict <- predict(no2.trendseason, newdata=next.7day)  no2.forecast <- forecast(no2.auto, h=7)  next.7day.prediction <- no2.predict + no2.forecast$mean  *# MSE:*  mean((next.7day.prediction-next.7day$time.no2)ˆ2) |

## [1] 34557.78

*#Figure 13:*

plot(ts(next.7day$time.no2),type='o',ylim=c(100,500),width=6, height=6)

## Warning in plot.window(xlim, ylim, log, ...): "width" is not a graphical ## parameter

## Warning in plot.window(xlim, ylim, log, ...): "height" is not a graphical ## parameter

## Warning in title(main = main, xlab = xlab, ylab = ylab, ...): "width" is not a ## graphical parameter

## Warning in title(main = main, xlab = xlab, ylab = ylab, ...): "height" is not a ## graphical parameter

## Warning in axis(1, ...): "width" is not a graphical parameter

## Warning in axis(1, ...): "height" is not a graphical parameter

## Warning in axis(2, ...): "width" is not a graphical parameter

## Warning in axis(2, ...): "height" is not a graphical parameter

## Warning in box(...): "width" is not a graphical parameter

## Warning in box(...): "height" is not a graphical parameter

lines(ts(next.7day.prediction),col='red',type='o')

lines(1:7, no2.predict + no2.forecast$lower[,2], col = "red", lty = "dashed") lines(1:7, no2.predict + no2.forecast$upper[,2], col = "red", lty = "dashed")

legend(1,500, legend = c("Actual", "Predicted"), lwd = 2, col = c("black", "red"))

ts(next.7day$time.no2)

1

2

3

4

5

6

7

100

200

300

400

500

Actual

Predicted

Time

# Question 2

1. **Here, we are setting the seed to 1 to begin our simulation. We simulateda model for a year of NO2 predictions using the auto regressive components, moving average components, and standard deviation components of our chosen autoregressive integrated moving average model. We then made a time variable that spanned an entire year and a whole data frame with time as the next year values. We then predicted the next year values using our trend season model. We combined the predicted next year NO2 values and the simulated autoregressive, moving average, and standard deviation components. Lastly, we converted the simulated dataset into a time series. The plot of the new simulated time series model looks similar to the observed time series model as seen in Figure 14.**

|  |
| --- |
| *#setting the seed* set.seed(1)  *#simulation of residuals*  auto.sim <- arima.sim(n=365, list(ar=c(no2.auto$coef[1],0), ma=c(no2.auto$coef[2])),  sd=sqrt(no2.auto$sigma2)) |

|  |
| --- |
| *# next year time variable and dataframe* next.yr.time <- c(1:365)  next.yr <- data.frame(time.no2 = next.yr.time)  next.yr.predictions <- predict(no2.trendseason, newdata=next.yr) no2.sim <- next.yr.predictions + auto.sim no2.sim.ts <- ts(no2.sim)  *#Figure 14*  autoplot(no2.sim.ts,xlab="Time",ylab="Simulated No2 concentrations",  width=6, height=6) |

100

200

300

0

100

200

300

Time

Simulated No2 concentrations

1. **Here, we started by creating a time variable which spans the length of a year.Then, we created a Day variable to create dummy variables to account for the weekly seasonality in the data. The contrasts of the Day variable shows that the base case is Friday. Lastly, we created a trend season model using our simulated time series data as our response variable, and the time and Day dummy variables as explanatory variables. The model is significant with a p-value of less than 2.2e16. The time variable is also significant with a coefficient of 0.24602 indicating that for every 1 increase in time, there is a 0.24602 increase in NO2 levels. In terms of the dummy variables, we found that Monday, Saturday,and Sunday are statistically significant with coefficients of -15.99891, -31.89066, and -38.74804, respectively. This indicates that the likelihood of NO2 levels being impacted by the day of week, specifically on Saturdays, Sundays, and Mondays may not be due to random chance.**

We also calculated percentage difference of the time coefficient between our observed trendseason model and our original trendseason model and found that there is a ~4.45% absolute difference.

|  |
| --- |
| *#Model the seasonality for simulated dataset* time.sim <- c(1:length(next.yr.predictions))  Day.sim <- rep(NA, length(time.sim))  Day.sim[which((time.sim %% 7) == 1)] <- "W"  Day.sim[which((time.sim %% 7) == 2)] <- "Th"  Day.sim[which((time.sim %% 7) == 3)] <- "Fri" Day.sim[which((time.sim %% 7) == 4)] <- "Sat"  Day.sim[which((time.sim %% 7) == 5)] <- "Sun"  Day.sim[which((time.sim %% 7) == 6)] <- "M"  Day.sim[which((time.sim %% 7) == 0)] <- "T"  Day.sim <- as.factor(Day.sim)  *#Trend season model of simulation*  no2.trendseason.sim<-lm(no2.sim.ts~time.sim + Day.sim) summary(no2.trendseason.sim) |

##

## Call:

## lm(formula = no2.sim.ts ~ time.sim + Day.sim)

##

## Residuals:

## Min 1Q Median 3Q Max

## -120.294 -24.907 0.866 24.137 89.080

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 133.4296 6.3263 21.091 < 2e-16 \*\*\* ## time.sim 0.2411 0.0190 12.689 < 2e-16 \*\*\* ## Day.simM -12.9412 7.5007 -1.725 0.08534 .

## Day.simSat -24.2375 7.5005 -3.231 0.00135 \*\*

|  |  |
| --- | --- |
| ## Day.simSun -37.1315 | 7.5006 -4.950 1.14e-06 \*\*\* |
| ## Day.simT 0.9409 | 7.5009 0.125 0.90025 |
| ## Day.simTh 3.4393 | 7.5005 0.459 0.64684 |
| ## Day.simW -10.9686 | 7.4651 -1.469 0.14263 |

## ---

## Signif. codes: 0 ’\*\*\*’ 0.001 ’\*\*’ 0.01 ’\*’ 0.05 ’.’ 0.1 ’ ’ 1

##

## Residual standard error: 38.25 on 357 degrees of freedom

## Multiple R-squared: 0.368, Adjusted R-squared: 0.3556

## F-statistic: 29.69 on 7 and 357 DF, p-value: < 2.2e-16

|  |
| --- |
| *#percent difference of time*  pct.diff <- ((no2.trendseason.sim$coefficients[2]-  no2.trendseason$coefficients[2])/  no2.trendseason$coefficients[2])\*100  pct.diff |

## time.sim

## -2.734943

**(c) Looking at the periodogram of the original time series, the most notable features are a large peak towards the beginning of the plot, the second largest peak around 0.15 freq, and a mini peak right before 0.3 freq. After the first large peak, all other peaks are considerably smaller and the data reaches a bit of plateau towards the end. Visually, the periodogram of the simulation has a higher quantity of noticeable peaks that are also much higher and sharper and does not emulate the plateau near the end of the graph. However, it does show a significant peak around 0.5 freq, similar to the original time series data, and another before 0.3 freq.**

*#periodogram for simulation*

pg.sim <- spec.pgram(no2.sim.ts,spans=9,demean=T,log='no', plot=F)

spec.sim <- data.frame(freq=pg.sim$freq, spec=pg.sim$spec) ggplot(spec.sim) + geom\_line(aes(x=freq,y=spec),width=6, height=6) +

ggtitle("Figure 15: Smooth Periodogram of Simulated No2")

## Warning: Ignoring unknown parameters: width, height

## Figure 15: Smooth Periodogram of Simulated No2

0

5000

10000

0.0

0.1

0.2

0.3

0.4

0.5

freq

spec

*#periodogram for observations*

pg.no2 <- spec.pgram(no2.ts,spans=9,demean=T,log='no',plot=F)

spec.no2 <- data.frame(freq=pg.no2$freq, spec=pg.no2$spec) ggplot(spec.no2) + geom\_line(aes(x=freq,y=spec),width=6, height=6) +

ggtitle("Figure 16: Smooth Periodogram of Observed No2")

## Warning: Ignoring unknown parameters: width, height

## Figure 16: Smooth Periodogram of Observed No2

0

5000

10000

15000

20000

25000

0.0

0.1

0.2

0.3

0.4

0.5

freq

spec

**(d) Based on our calculations, the mean of the observed time series dataset is 164.80, while the predicted dataset mean is 163.38, resulting in a -0.086% difference. Thus, the prediction’s estimation of mean is relatively accurate. As for the variance, the observed data’s variance is 2616.039, while the simulated dataset’s variance is 2076.232; the percent difference between the observed and simulated variance is -20.63%, meaning the model was less accurate at reproducing the variance of the original dataset. This is a weak point in our model that we would continue to dissect and address in future iterations of analysis.**

*#mean of observed* mean.no2 <- mean(no2.ts)

*#mean of simulated* mean.sim <- mean(no2.sim.ts)

*#variance of observed* var.no2 <- var(no2.ts)

*#variace of simulated* var.sim <- var(no2.sim.ts) mean.no2

## [1] 164.7976

mean.sim

## [1] 165.9962

var.no2

## [1] 2616.039

var.sim

## [1] 2269.759

*#pct diffs between means* (mean.sim-mean.no2)/(mean.no2)\*100

## [1] 0.7273139

*#pct diffs between vars* ((var.sim-var.no2)/var.no2)\*100

## [1] -13.23678

**(e) The ACF of the observed dataset displays a sinusoidal decay pattern with a peak every 7 days. The ACF of the simulated data somewhat displays this pattern with peaks occurring every 7 days that have a greater amplitude than those in the observed dataset. The PACF of the observed dataset has roughly 6 significant lags around 2,3,5,6,7 and become insignificant around lag 8, whereas the PACF of the simulated dataset has roughly 3 significant lags around 6,7,13 (which is beyond 8). Thus, the simulation was only able to reproduce some components of the autocorrelation of the ovserved values, but requires improvement in other areas. In future work, we would look further into exploring the autoregressive and moving average components of our data to produce more accurate results.**

*#acf of simulation* acf.sim <- acf(no2.sim.ts)

## Series no2.sim.ts

0

5

10

15

20

25

0.0

0.2

0.4

0.6

0.8

1.0

ACF

Lag

*#acf of obsverations* acf.obs <- acf(no2.ts)

## Series no2.ts

0

5

10

15

20

25

0.0

0.2

0.4

0.6

0.8

1.0

ACF

Lag

*#pacf of simulation* pacf.sim <- pacf(no2.sim.ts)

## Series no2.sim.ts

5

10

15

20

25

0.0

0.2

0.4

0.6

Partial ACF

Lag

*#pacf of obsverations* pacf.obs <- pacf(no2.ts)

## Series no2.ts

5

10

15

20

25

0.0

0.2

0.4

0.6

Partial ACF

Lag

Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.