Project 2

Shayne Cassidy, Sarah Nelson, Sarah Winston Nathan

12-6-2019

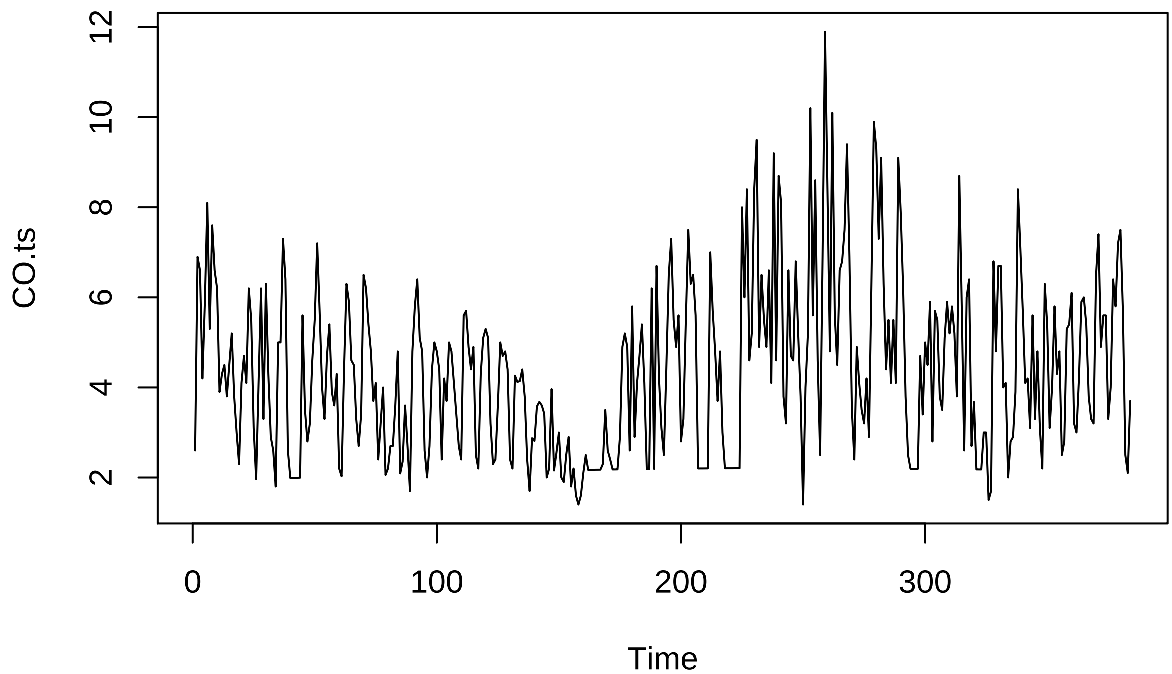
Load data and impute missing values

```
setwd(datadir)
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(1:3)],'~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)

# remove last 7 days
dailyAQ <- dailyAQ[1:(dim(dailyAQ)[1]-7),]
```

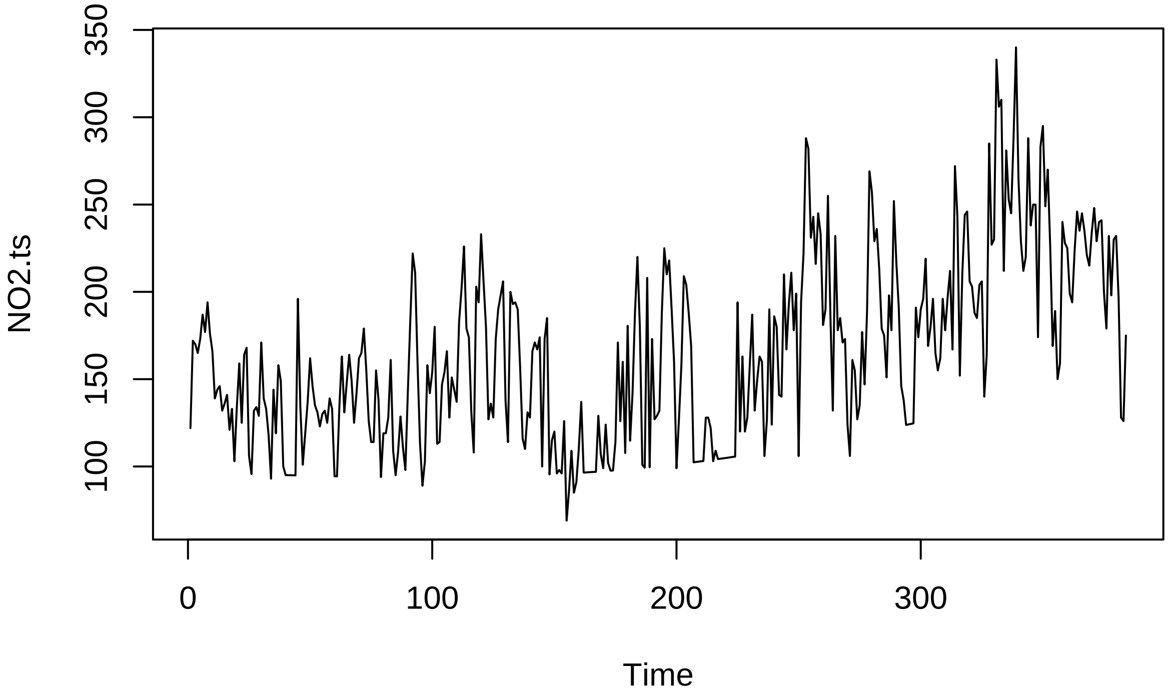
Part 1: Building Univariate Time Series Models

```
AQ.CO <- dailyAQ$CO.GT.
#AQ.CO <- AQdata$CO.GT.
AQ.NO2 <- dailyAQ$NO2.GT.
#AQ.NO2<-AQdata$NO2.GT.

CO.ts<-ts(AQ.CO)
NO2.ts<-ts(AQ.NO2)
```

```r
plot(CO.ts)
```

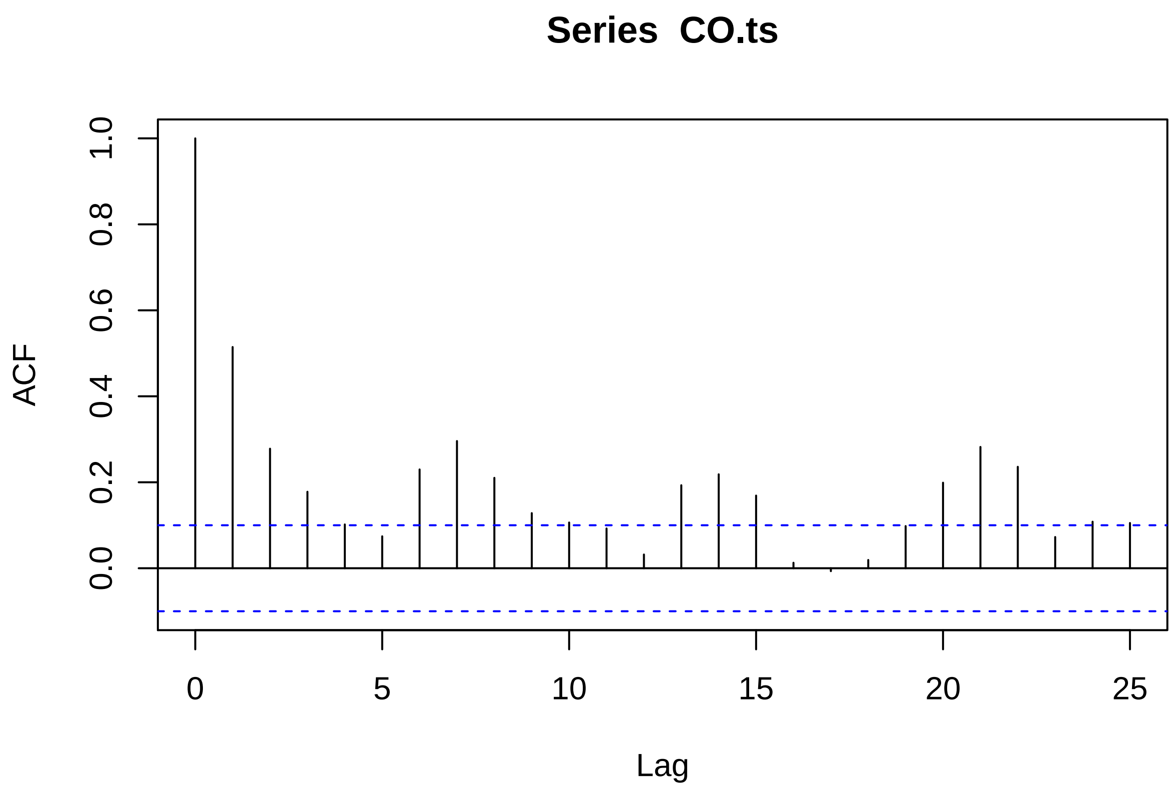



```r
plot(NO2.ts)
```

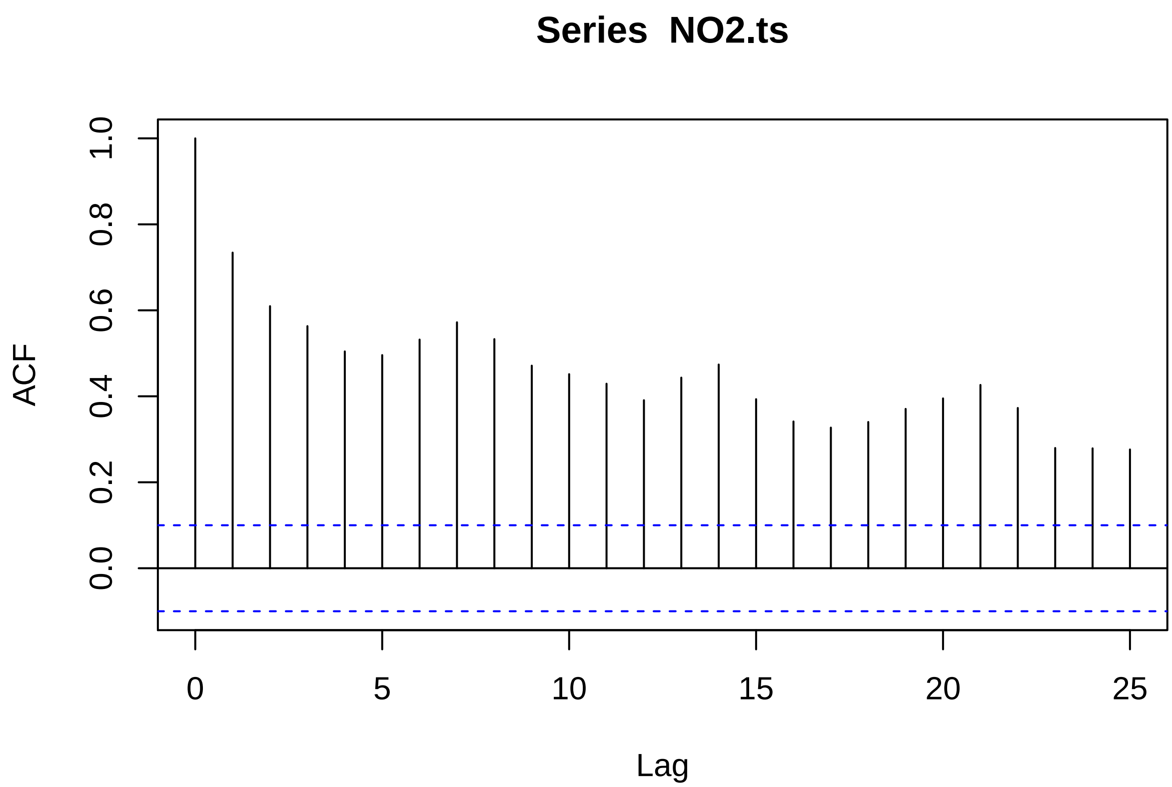


Part A: Seasonality

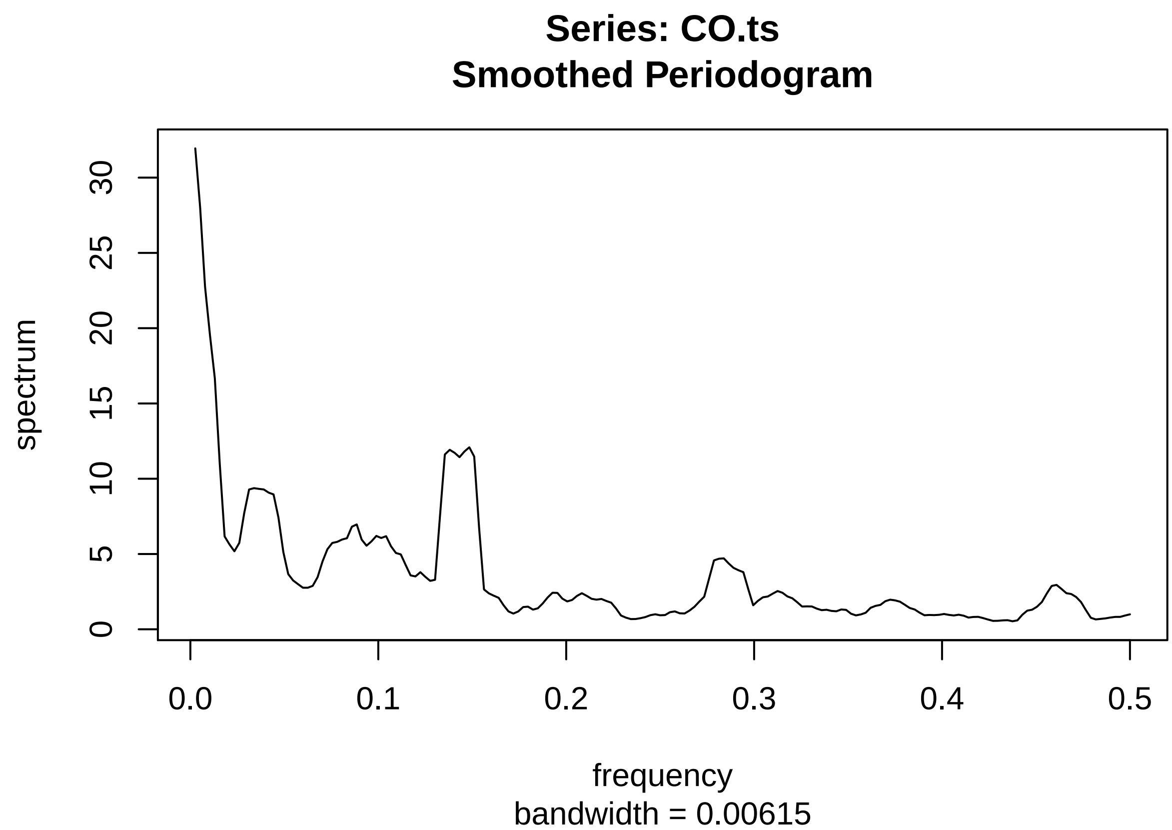
```
acf(CO.ts)
```



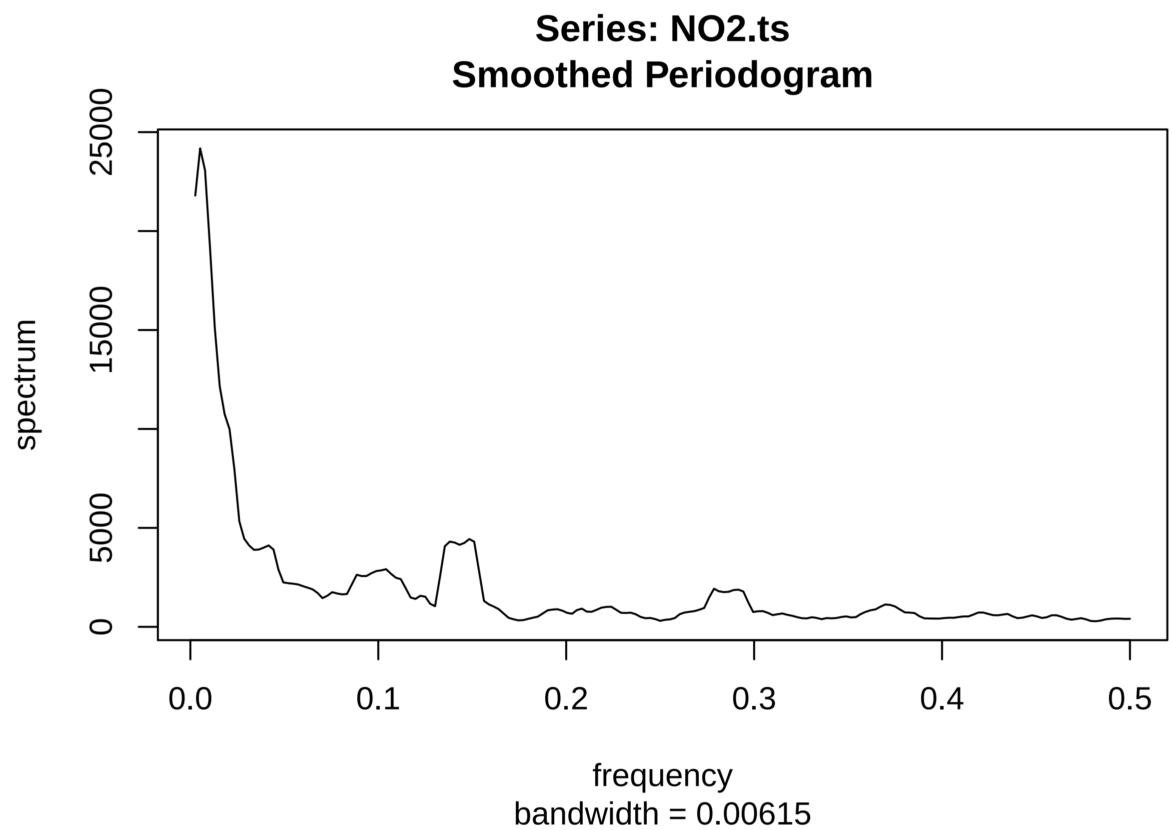

```
acf(NO2.ts)
```




```
pg.CO <- spec.pgram(CO.ts,spans=9,demean=T,log='no')
```




```
pg.NO2 <- spec.pgram(NO2.ts,spans=9,demean=T,log='no')
```



The spikes in both periodagrams at repeated frequencies indicates seasonality present.

```r
# What are the periods of the next biggest peaks?
# sort spectrum from largest to smallest and find index
sorted.spec <- sort(pg.CO$spec, decreasing=T, index.return=T)
names(sorted.spec)
```

```
## [1] "x"  "ix"
```

```r
# corresponding periods
sorted.omegas <- pg.NO2$freq[sorted.spec$ix]
sorted.Ts <- 1/pg.NO2$freq[sorted.spec$ix]

# look at first 20
sorted.omegas[1:20]
```

```
##  [1] 0.002604167 0.005208333 0.007812500 0.010416667 0.013020833
##  [6] 0.148437500 0.138020833 0.145833333 0.140625000 0.135416667
## [11] 0.151041667 0.143229167 0.015625000 0.033854167 0.036458333
## [16] 0.039062500 0.031250000 0.041666667 0.044270833 0.028645833
```

```r
sorted.Ts[1:20]
```

```
##  [1] 384.000000 192.000000 128.000000  96.000000  76.800000   6.736842
##  [7]   7.245283   6.857143   7.111111   7.384615   6.620690   6.981818
```

```
## [13]  64.000000  29.538462  27.428571  25.600000  32.000000  24.000000
## [19]  22.588235  34.909091
```

```
# evens out around 7
period<-7
```

Part B: Trends

Build a new model, CO.trend which predicts CO.ts based on the time variable

```
time<-c(1:(length(CO.ts)))
CO.trend<-lm(CO.ts ~ time)
NO2.trend<-lm(NO2.ts ~ time)

summary(CO.trend)
```

```
##
## Call:
## lm(formula = CO.ts ~ time)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.2485 -1.6980 -0.0525  1.0863  7.3442
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.810929   0.192140  19.834  < 2e-16 ***
## time        0.002876   0.000865   3.325  0.00097 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.879 on 382 degrees of freedom
## Multiple R-squared:  0.02813,    Adjusted R-squared:  0.02558
## F-statistic: 11.06 on 1 and 382 DF,  p-value: 0.0009695
```

```
summary(NO2.trend)
```

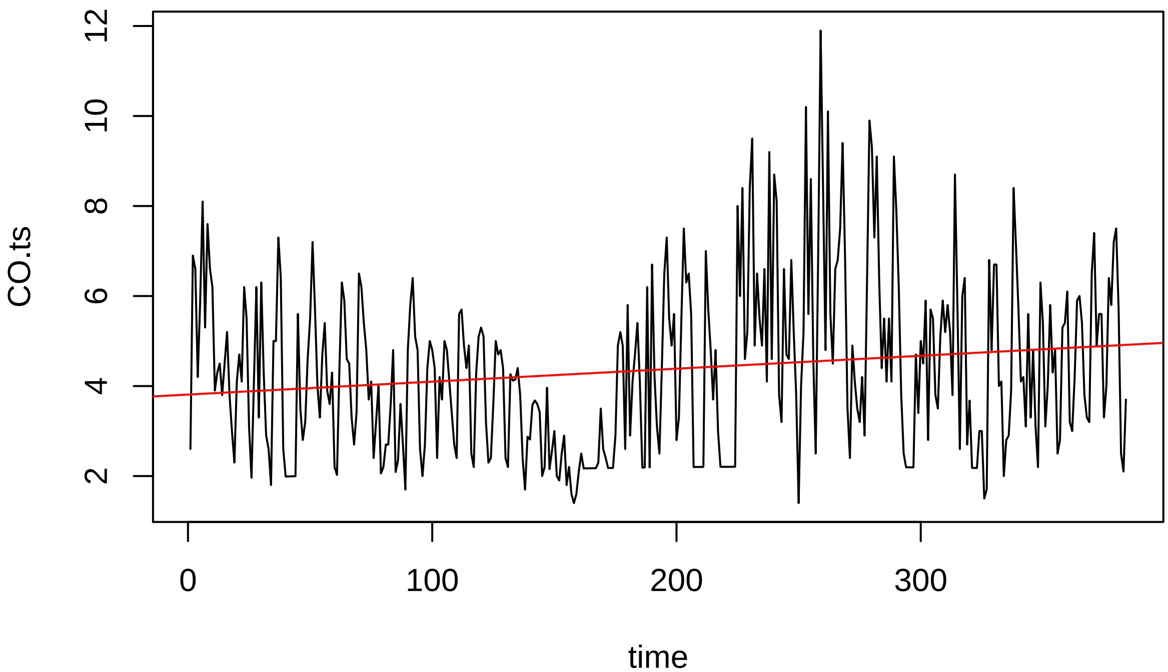
```
##
## Call:
## lm(formula = NO2.ts ~ time)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -87.389 -34.365   2.159  27.847 137.895
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 115.16473    4.40111   26.17   <2e-16 ***
## time          0.25646    0.01981   12.94   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 43.04 on 382 degrees of freedom
## Multiple R-squared:  0.3049, Adjusted R-squared:  0.3031
## F-statistic: 167.6 on 1 and 382 DF,  p-value: < 2.2e-16
```

Here we built two new models, CO.trend and No2.trend, that both model the trend components. For CO.trend, the p-value is 0.00097, and for NO2.trend, the p-value is <2.2e-16. Therefore, the trend component is significant in both models and must be considered.

Plot CO.trend model

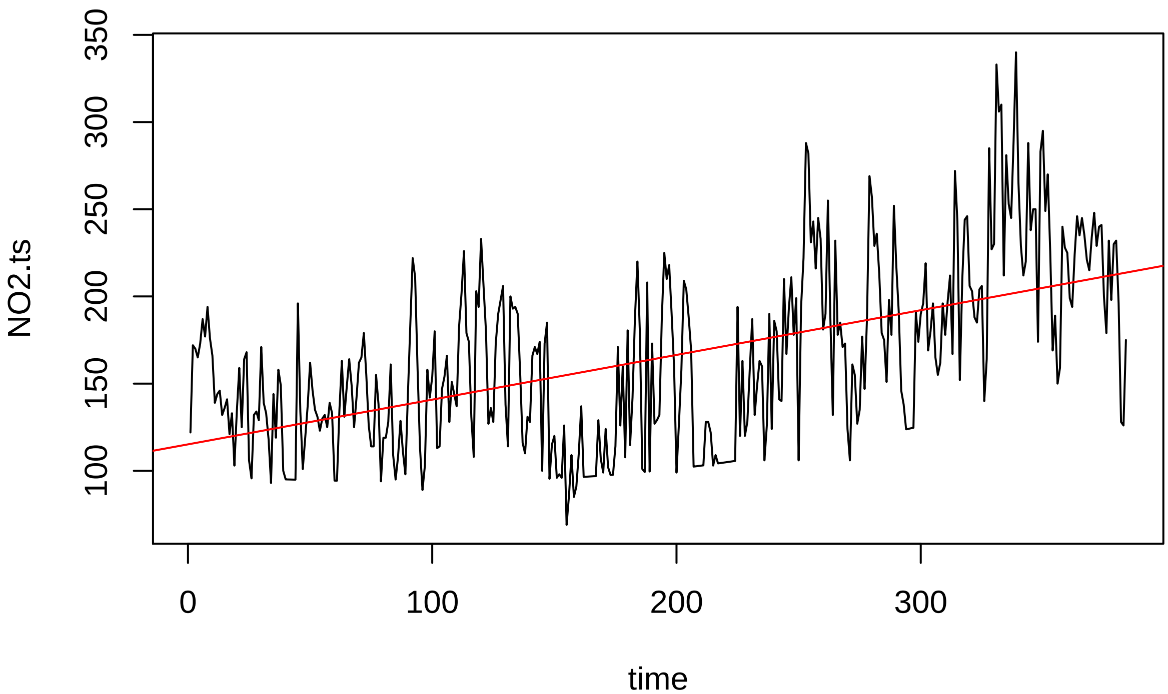
```
{plot(time, CO.ts, type = "l")
abline(CO.trend, col = "red")}
```



As seen in the plot of the CO.trend model, we can see that there is a clear upward trend line, which supports the results of our statistical test. The adjusted R^2 for the model CO.trend is 0.02558.

Plot NO2.trend model

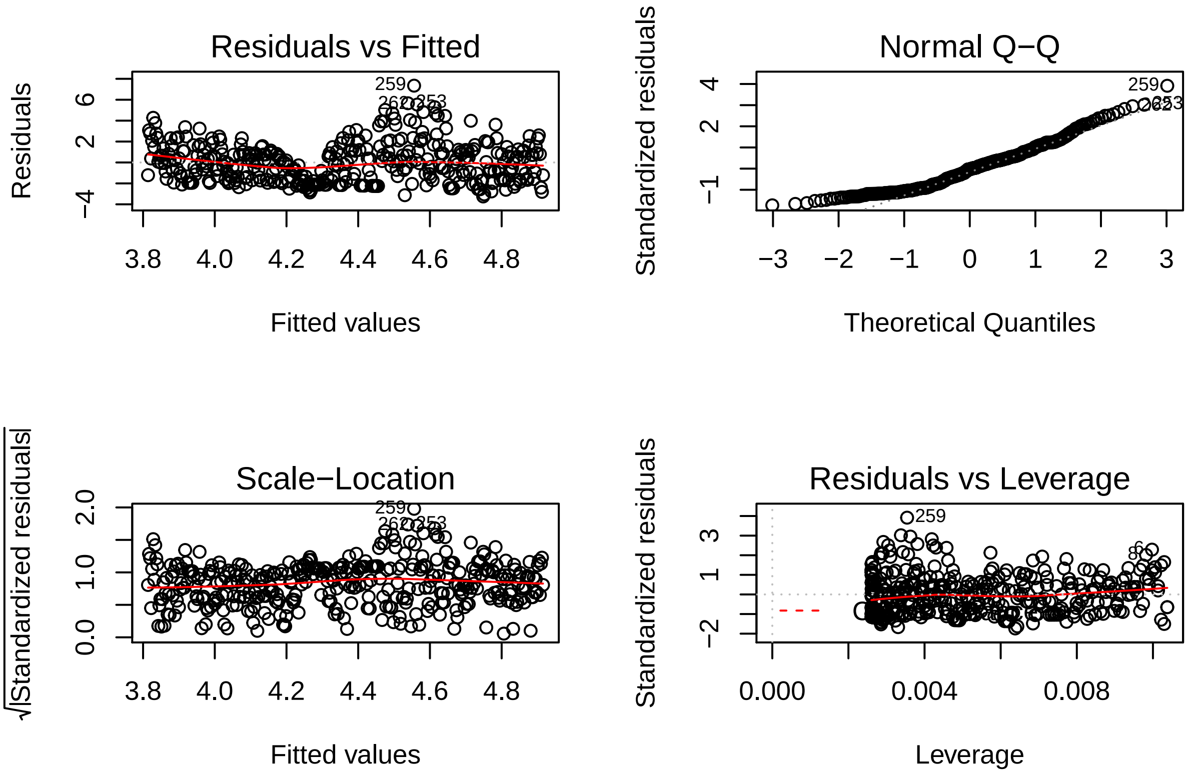
```
{plot(time, NO2.ts, type = "l")
abline(NO2.trend, col = "red")}
```



As seen in the plot of the NO2.trend model, we can see that there is a clear upward trend line. From the naked eye, the slope seems more drastic than with the trend line from CO.trend model. This supports the results of our statistical test. The adjusted R^2 for the model NO2.trend is 0.3031.

Model diagnostics for CO.trend

```
par(mfrow=c(2,2))
plot(CO.trend, labels.id = NULL)
```



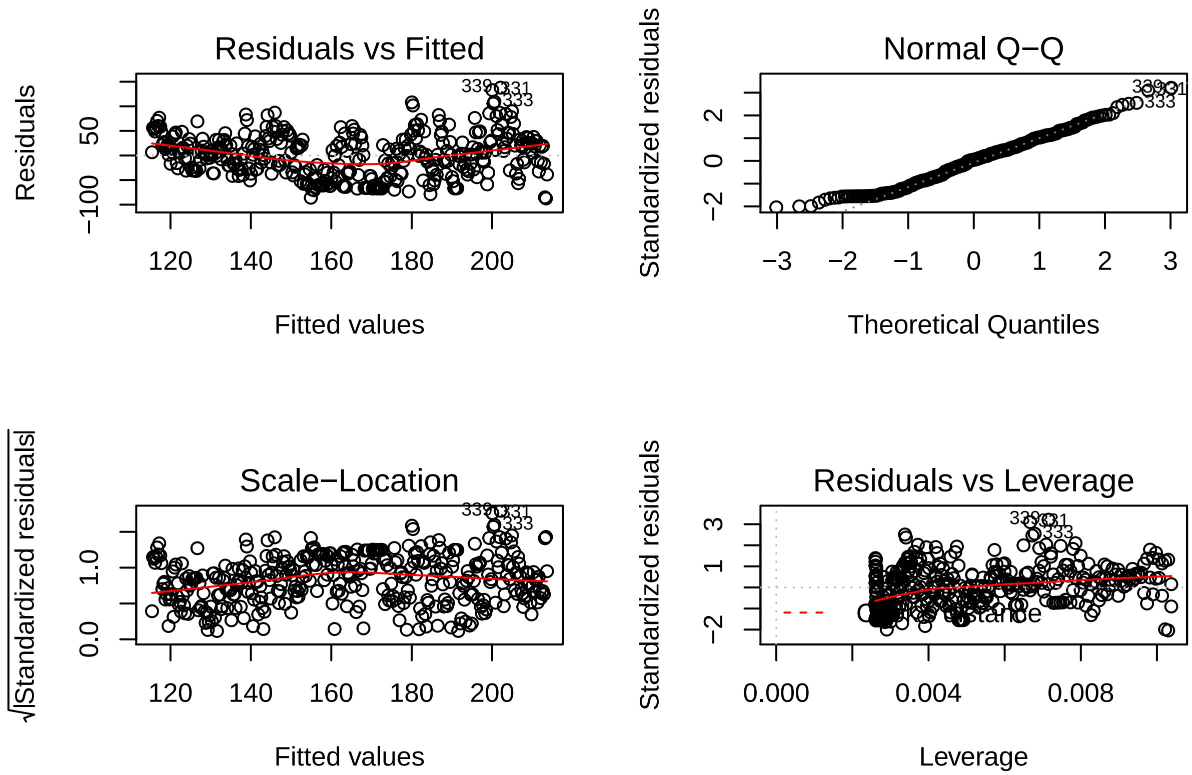

```
par(mfrow=c(1,1))
```

Residuals versus fitted plot: Does not violate assumptions. The mean is around zero and there seems to be constant variance. There are a few outliers. Q-Q plot: The fit to the line is fairly solid, thus no drastic violation of assumptions. The Q-Q plot could be improved. Scale-location: Does not violate assumptions. The mean is about zero and there seems to be constant variance. There are a few outliers. Residuals versus leverage: No clear influential points with regards to Cook's distance.

The diagnostic plots do not indicate any strong violations of assumptions, so it does not appear that we need to perform any type of transformation (i.e. Log transform).

Model diagnostics for NO2.trend

```
par(mfrow=c(2,2))
plot(NO2.trend, labels.id = NULL)
```




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

Residuals versus fitted plot: Does not violate assumptions. The mean is about zero and there seems to be constant variance. Q-Q plot: The fit to the line is fairly solid, thus no drastic violation of assumptions. As seen by the naked eye, the Q-Q plot for N02.trend is a little better than that of CO.trend. Scale-location: Does not violate assumptions. The mean is around 0.75 and there seems to be a constant variance around this mean. While we would prefer for the mean to be closer to zero, paired with the other diagnostic plots we do not feel like this causes a violation of assumptions. There are a few outliers. Residuals versus leverage: No clear influential points with regards to Cook's distance.

Based on these diagnostics plots, it does not appear that we need to perform any type of transformation (i.e. Log transform).

Add seasonality component to CO.trend

Because the seasonality component was significant, we added a seasonality component to CO.trend. We decided to use a period of 7 because the peak frequency of the periodogram is about 0.14, and $1/0.14 = 7$. This implies a weekly period.

```r
CO.trend.seasonal <- lm(CO.ts[time] ~ time + sin(2*pi*time/7) + cos(2*pi*time/7))
summary(CO.trend.seasonal)
```

```
##
## Call:
```

```
## lm(formula = CO.ts[time] ~ time + sin(2 * pi * time/7) + cos(2 *
##     pi * time/7))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4604 -1.1866 -0.1247  1.0272  6.9821
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          3.7979449  0.1805339  21.037  < 2e-16 ***
## time                 0.0029483  0.0008127   3.628 0.000325 ***
## sin(2 * pi * time/7) 0.8531164  0.1272445   6.705 7.33e-11 ***
## cos(2 * pi * time/7) 0.3563039  0.1275659   2.793 0.005485 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.765 on 380 degrees of freedom
## Multiple R-squared:  0.1466, Adjusted R-squared:  0.1399
## F-statistic: 21.76 on 3 and 380 DF,  p-value: 5.027e-13
```

The p-value for this model is 5.027e-13. The adjusted R^2 for the model CO.trend.seasonal is 0.1399.

Add seasonality component to NO2.trend

Because the seasonality component was significant, we added a seasonality component to N02.trend. We decided to use a period of 7 because the peak frequency of the periodogram is about 0.14, and 1/0.14 = 7. This implies a weekly period.

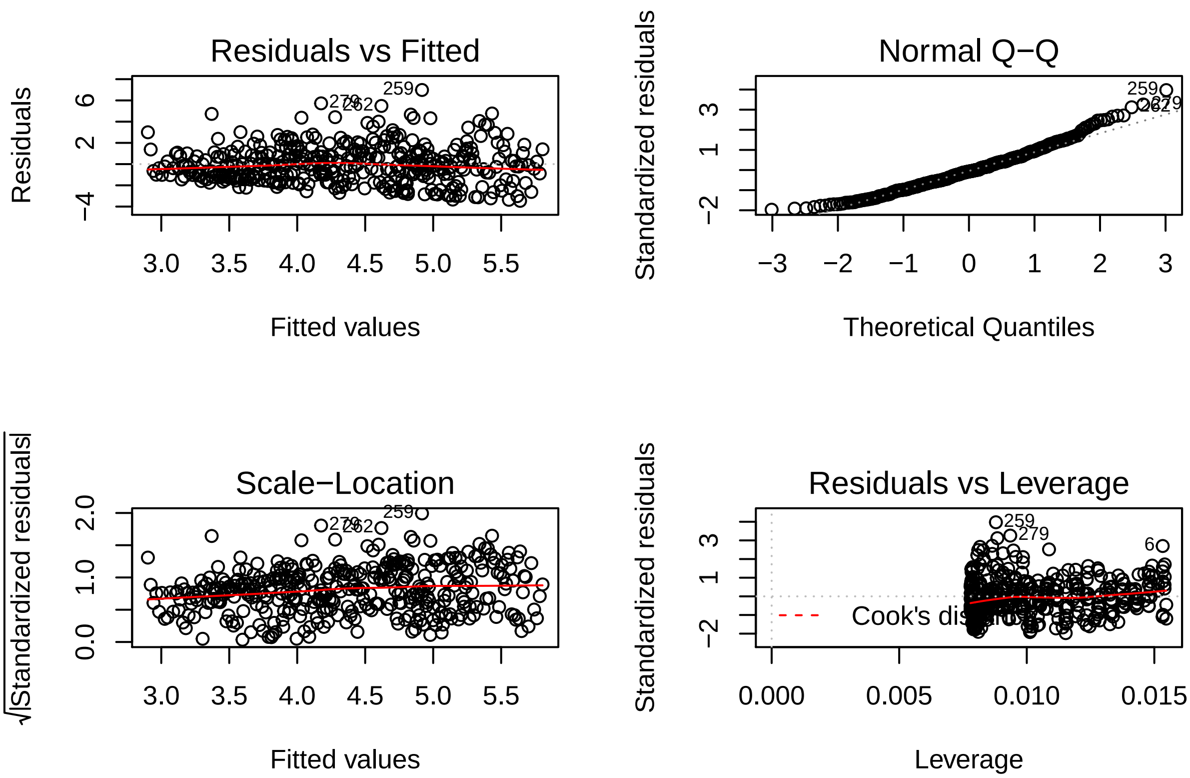
```r
NO2.trend.seasonal <- lm(NO2.ts[time] ~ time + sin(2*pi*time/7) + cos(2*pi*time/7))
summary(NO2.trend.seasonal)
```

```
##
## Call:
## lm(formula = NO2.ts[time] ~ time + sin(2 * pi * time/7) + cos(2 *
##     pi * time/7))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -101.641  -28.675    1.226   26.385  135.816
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          114.9221     4.2424  27.089  < 2e-16 ***
## time                   0.2578     0.0191  13.498  < 2e-16 ***
## sin(2 * pi * time/7)  15.7600     2.9902   5.271 2.28e-07 ***
## cos(2 * pi * time/7)   5.5152     2.9977   1.840   0.0666 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 41.48 on 380 degrees of freedom
## Multiple R-squared:  0.3576, Adjusted R-squared:  0.3525
## F-statistic:  70.5 on 3 and 380 DF,  p-value: < 2.2e-16
```

The p-value for this model is 2.2e-16. The adjusted R^2 for the model N02.trend.seasonal is 0.3525

Model diagnostics for CO.trend.seasonal

```r
par(mfrow=c(2,2))
plot(CO.trend.seasonal, labels.id = NULL)
```

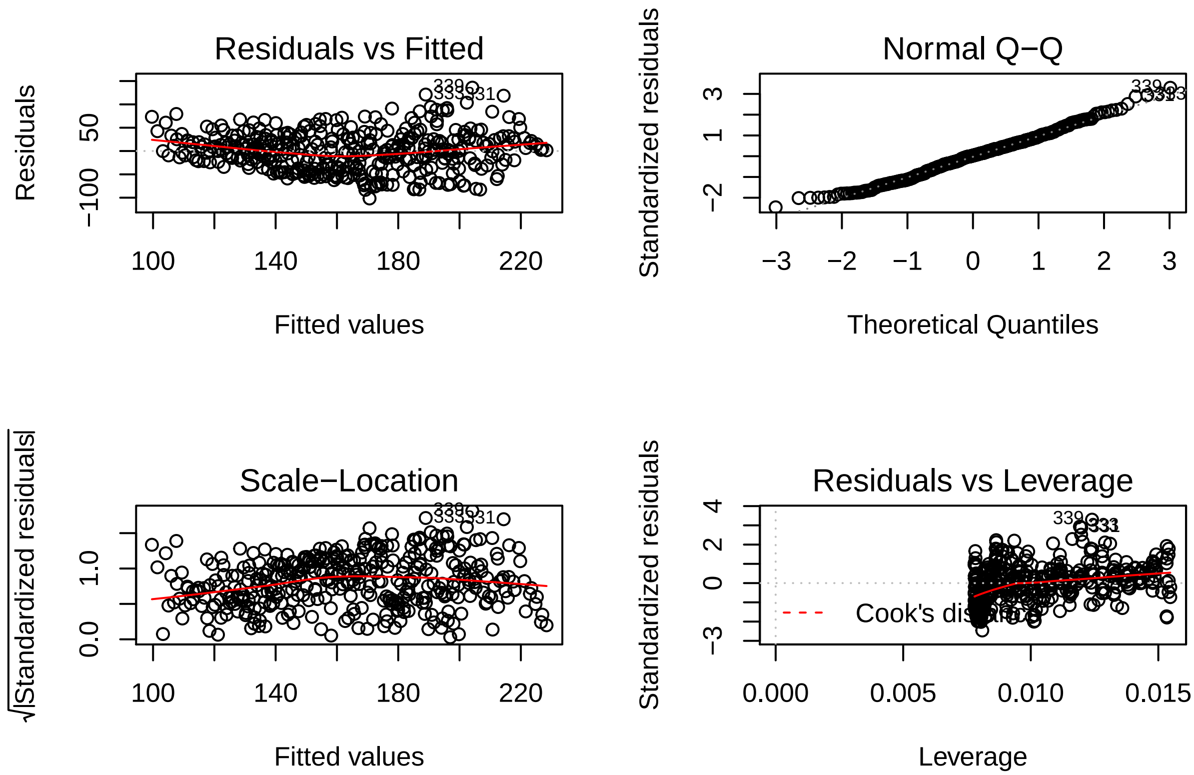



```r
par(mfrow=c(1,1))
```

Residuals versus fitted plot: Does not violate assumptions. The mean is about zero and there seems to be constant variance. There are a few outliers. Q-Q plot: The fit to the line is fairly solid, thus no drastic violation of assumptions. Scale-location: Does not violate assumptions. The mean is about zero and there seems to be constant variance. Residuals versus leverage: No clear influential points with regards to Cook's distance. The spread of points above and below the mean line in the residuals versus fitted plot and scale-location plots have improved from the CO.trend model.

Model diagnostics for NO2.trend.seasonal

```r
par(mfrow=c(2,2))
plot(NO2.trend.seasonal, labels.id = NULL)
```




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

Residuals versus fitted plot: Does not violate assumptions. The mean is about zero and there seems to be constant variance. Q-Q plot: The fit to the line is fairly solid, thus no drastic violation of assumptions. The Q-Q plot could be improved a bit. Scale-location: Does not violate assumptions. The mean is about zero and there seems to be constant variance. There are a few outliers. Residuals versus leverage: No clear influential points with regards to Cook's distance.

Both the CO.trend.seasonal model and the NO2.trend.seasonal model demonstrate good diagnostics which do not violate assumptions, so we will not do a transformation of either model.

Part C: Auto-Regressive and Moving Average

#Get the residuals from the CO.trend.seasonal model above and store in e.ts:

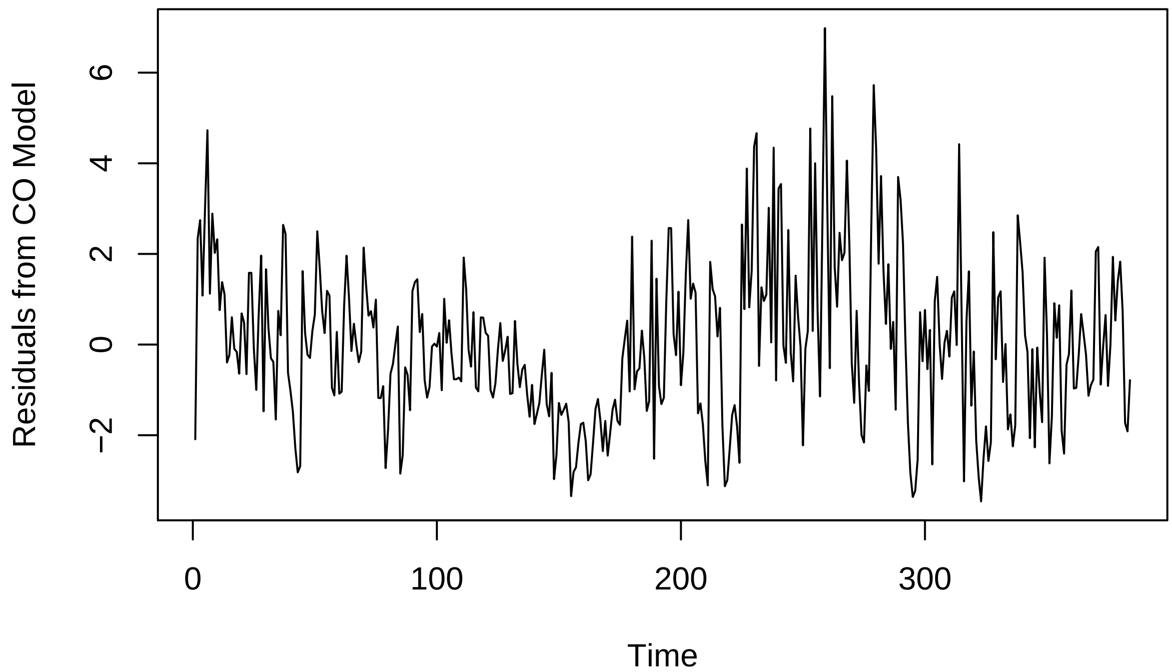
```r
e.ts.CO<-ts(CO.trend.seasonal$residuals)
```

#Get the residuals from the NO2.trend.seasonal model above and store in e.ts:

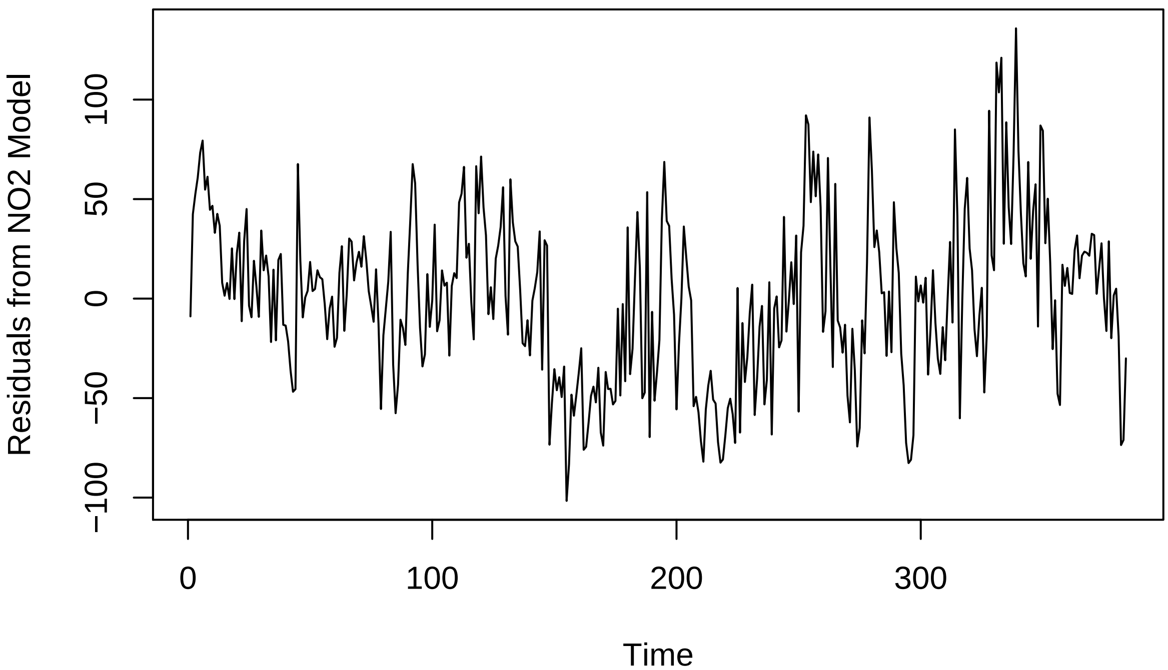
```r
e.ts.NO2<-ts(NO2.trend.seasonal$residuals)
```

#Plot the residuals for the CO.trend.seasonal model NO2.trend.seasonal

```r
plot(e.ts.CO, ylab = "Residuals from CO Model")
```

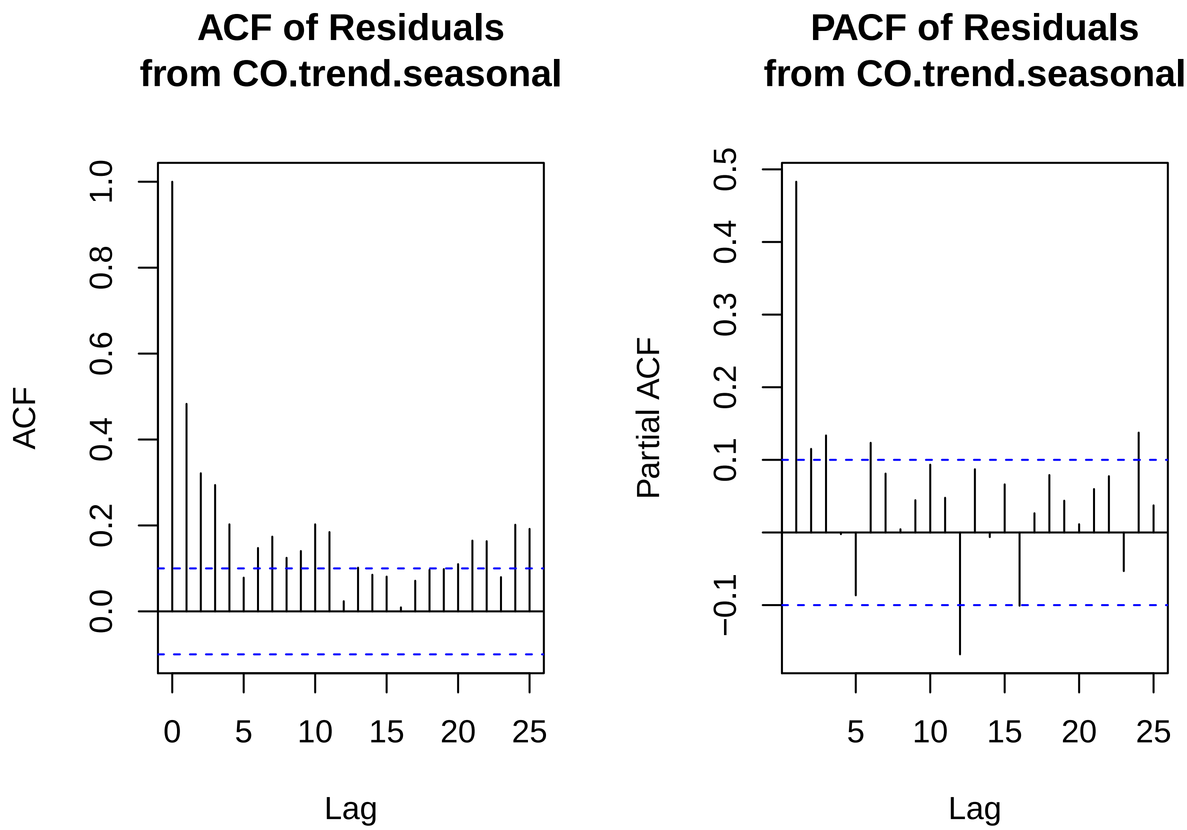



```r
plot(e.ts.NO2, ylab = "Residuals from NO2 Model")
```



Plot the autocorrelation (ACF) and partial autocorrelation (PACF) of the residuals of CO.trend.seasonal

```
par(mfrow=c(1,2))
acf(e.ts.CO, main="ACF of Residuals\nfrom CO.trend.seasonal")
pacf(e.ts.CO,main="PACF of Residuals\nfrom CO.trend.seasonal")
```

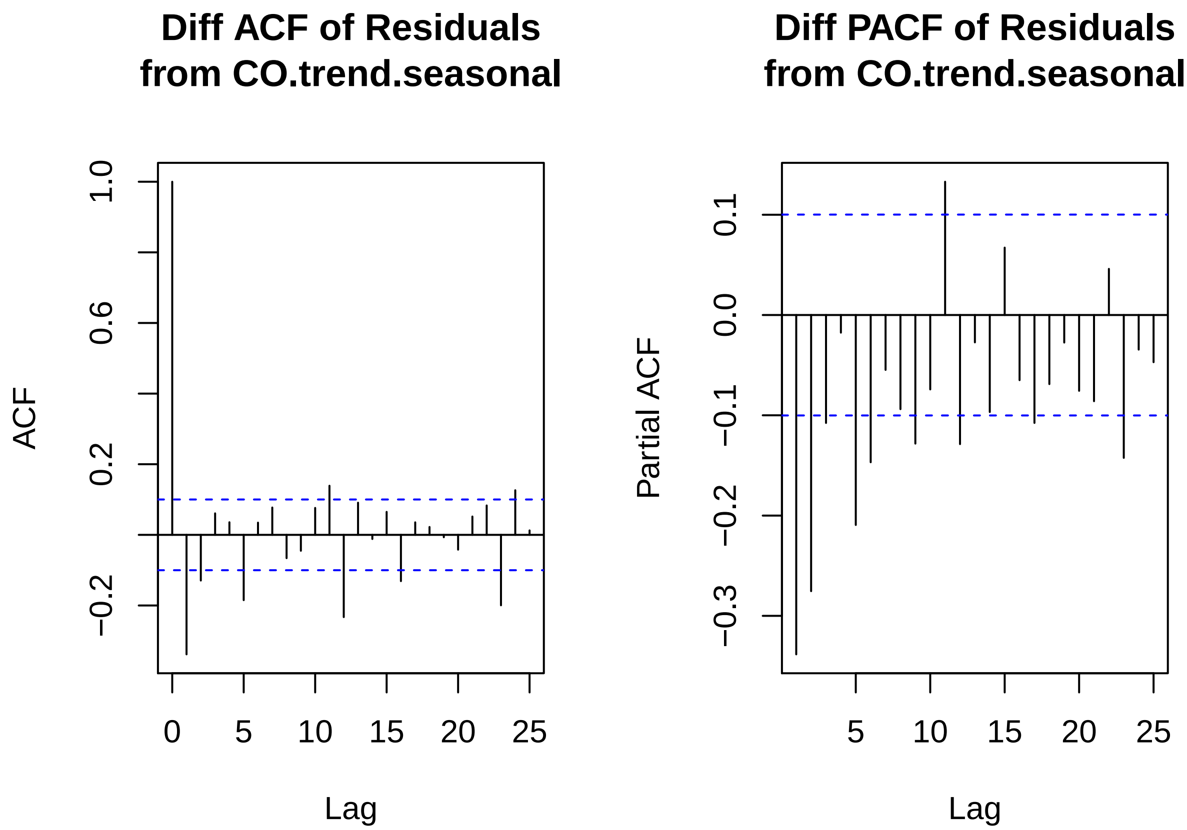



```
par(mfrow=c(1,1))
```

The ACF plot for the residuals of the CO.trend.seasonal shows sinusoidal decay. However, it cuts off at a very high lag, thus we are going to test various moving average components. The PACF plot for the residuals of the CO.trend.seasonal shows sinusoidal decay. However, it cuts off at a very high lag, thus we are going to test various autoregressive components. Because the ACF and PACF both show sinusoidal decay, we will also test some ARMA models with both autoregressive and moving average components. Then we will calculate AIC values to assess several model choices.

Do we need to consider a first order difference of our residuals?

```
par(mfrow=c(1,2))
acf(diff(e.ts.CO), main="Diff ACF of Residuals\nfrom CO.trend.seasonal")
pacf(diff(e.ts.CO),main="Diff PACF of Residuals\nfrom CO.trend.seasonal")
```

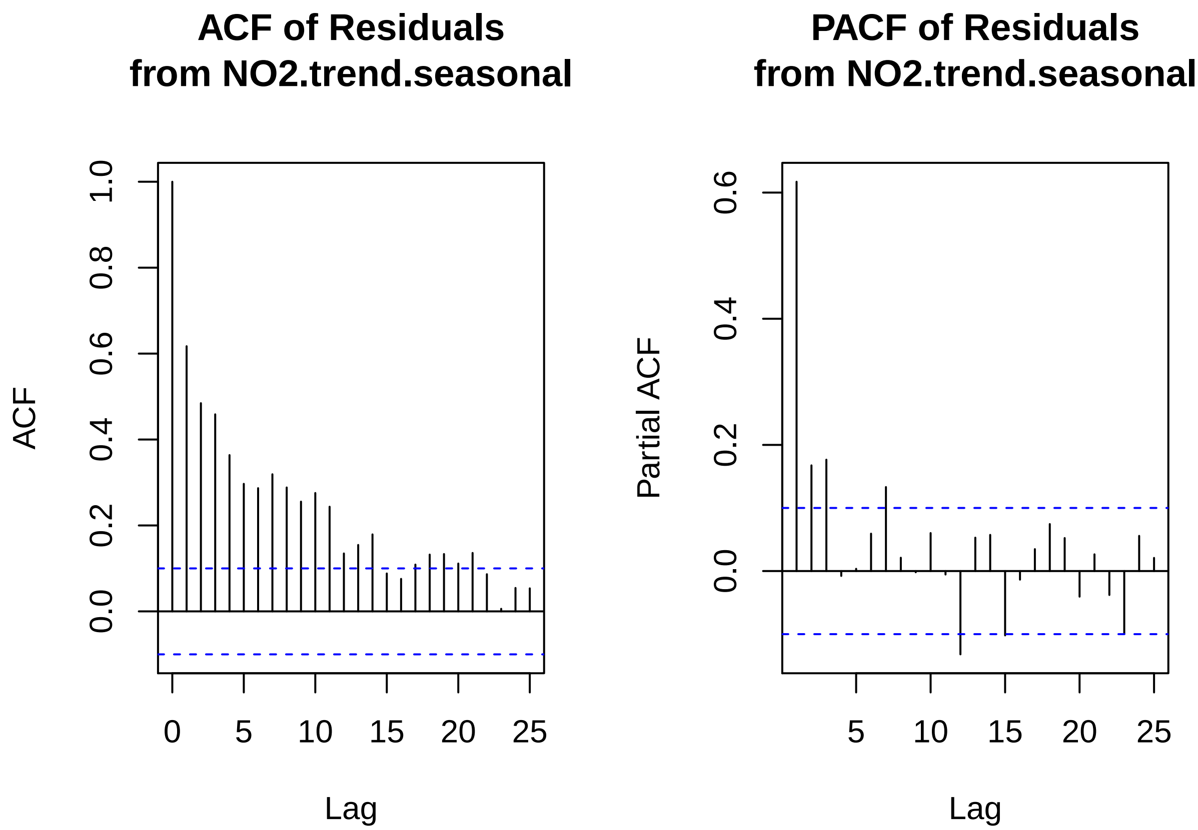



```
par(mfrow=c(1,1))
```

No, we do not need to consider a first order difference the residuals of the CO.trend.seasonal model because the ACF shows sinusoidal decay that does not cut off, and the differentiated ACF does not improve this. Thus, the value of d is 0.

Plot the autocorrelation (ACF) and partial autocorrelation (PACF) of the residuals of NO2.trend.seasonal

```
par(mfrow=c(1,2))
acf(e.ts.NO2, main="ACF of Residuals\nfrom NO2.trend.seasonal")
pacf(e.ts.NO2,main="PACF of Residuals\nfrom NO2.trend.seasonal")
```

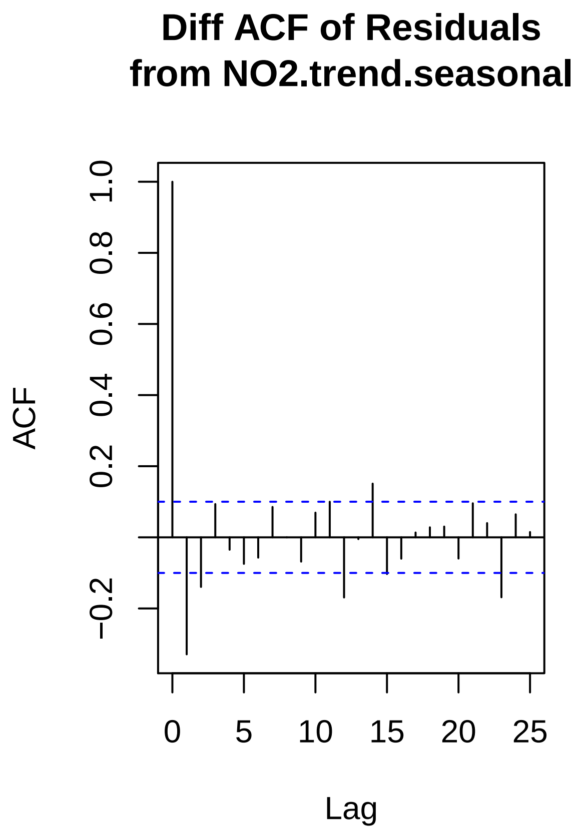


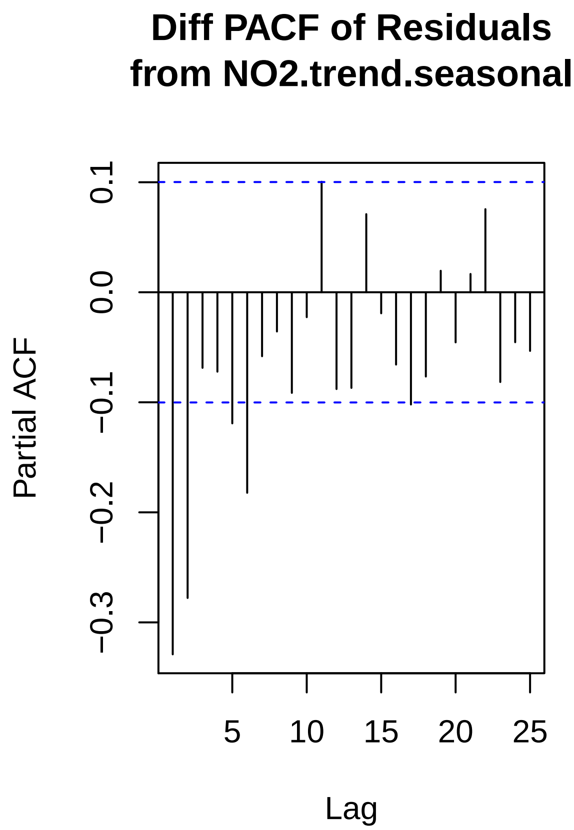

```
par(mfrow=c(1,1))
```

The ACF plot for the residuals of the NO2.trend.seasonal shows sinusoidal decay. However, it cuts off at a very high lag, thus we are going to test various moving average components. The PACF plot for the residuals of the NO2.trend.seasonal shows sinusoidal decay. However, it also cuts off at a very high lag, thus we are going to test various autoregressive components. Because the ACF and PACF both show sinusoidal decay, we will also test some ARMA models with both autoregressive and moving average components. Then we will calculate AIC values to assess several model choices.

Do we need to consider a first order difference of our residuals?

```
par(mfrow=c(1,2))
acf(diff(e.ts.NO2), main="Diff ACF of Residuals\nfrom NO2.trend.seasonal")
pacf(diff(e.ts.NO2),main="Diff PACF of Residuals\nfrom NO2.trend.seasonal")
```






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

Both plots show sinusoidal decay, which points to using an ARMA model.

No, we do not need to consider a first order difference the residuals of the NO2.trend.seasonal model because the ACF shows sinusoidal decay that does not cut off, and the differentiated ACF does not improve this. Thus, the value of d is 0.

Modeling e.ts.CO

Now we will try out some models for e.ts.CO using various p and q values

ar(1) p=1

```
CO.ar1 <- arima(e.ts.CO, order=c(1,0,0), include.mean=FALSE)
summary(CO.ar1)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(1, 0, 0), include.mean = FALSE)
##
## Coefficients:
##          ar1
##       0.4837
```

```
## s.e.  0.0446
##
## sigma^2 estimated as 2.361:  log likelihood = -709.99,  aic = 1423.98
##
## Training set error measures:
##                        ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.0003100464 1.536712 1.181932 146.0159 267.2095 0.9043589
##                     ACF1
## Training set -0.05561519
```

AIC = 1423.98

ar(2) p=2

```
CO.ar2 <- arima(e.ts.CO, order=c(2,0,0), include.mean=FALSE)
summary(CO.ar2)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(2, 0, 0), include.mean = FALSE)
##
## Coefficients:
##          ar1     ar2
##       0.4281  0.1137
## s.e.  0.0509  0.0511
##
## sigma^2 estimated as 2.331:  log likelihood = -707.52,  aic = 1421.05
##
## Training set error measures:
##                        ME    RMSE      MAE      MPE     MAPE      MASE
## Training set -0.0007252218 1.52683 1.176321 147.9407 265.1116 0.9000656
##                     ACF1
## Training set -0.01513917
```

AIC = 1421.05

ar(3) p=3

```
CO.ar3 <- arima(e.ts.CO, order=c(3,0,0), include.mean=FALSE)
summary(CO.ar3)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(3, 0, 0), include.mean = FALSE)
##
## Coefficients:
##          ar1     ar2     ar3
##       0.4120  0.0565  0.1344
## s.e.  0.0508  0.0551  0.0510
##
## sigma^2 estimated as 2.289:  log likelihood = -704.08,  aic = 1416.17
##
## Training set error measures:
```

```
##                           ME     RMSE     MAE      MPE     MAPE      MASE
## Training set -0.001624096 1.513104 1.15694 177.2785 280.2686 0.8852366
##                        ACF1
## Training set 0.001369499
```

AIC = 1416.17

ma(1) p=0, q=1

```r
CO.ma1 <- arima(e.ts.CO, order=c(0,0,1), include.mean=FALSE)
summary(CO.ma1)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(0, 0, 1), include.mean = FALSE)
##
## Coefficients:
##          ma1
##       0.3947
## s.e.  0.0405
##
## sigma^2 estimated as 2.524:  log likelihood = -722.69,  aic = 1449.39
##
## Training set error measures:
##                            ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -9.756774e-05 1.588605 1.231327 130.6677 209.6848 0.9421537
##                       ACF1
## Training set 0.08257549
```

AIC = 1449.39

ma(2) p=0, q=2

```r
CO.ma2 <- arima(e.ts.CO, order=c(0,0,2), include.mean=FALSE)
summary(CO.ma2)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(0, 0, 2), include.mean = FALSE)
##
## Coefficients:
##          ma1     ma2
##       0.4323  0.1491
## s.e.  0.0511  0.0415
##
## sigma^2 estimated as 2.443:  log likelihood = -716.49,  aic = 1438.97
##
## Training set error measures:
##                          ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 1.12221e-05 1.563074 1.205482 141.0392 246.6945 0.9223784
##                       ACF1
## Training set 0.02334346
```

AIC = 1438.97

ma(3) p=0, q=3

```
CO.ma3 <- arima(e.ts.CO, order=c(0,0,3), include.mean=FALSE)
summary(CO.ma3)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(0, 0, 3), include.mean = FALSE)
##
## Coefficients:
##          ma1     ma2     ma3
##       0.3772  0.2265  0.2156
## s.e.  0.0503  0.0475  0.0517
##
## sigma^2 estimated as 2.351:  log likelihood = -709.12,  aic = 1426.24
##
## Training set error measures:
##                         ME     RMSE      MAE    MPE    MAPE      MASE
## Training set -0.0007200315 1.533204 1.193358 96.042 195.776 0.9131019
##                    ACF1
## Training set 0.03672645
```

AIC = 1426.24

arma(1,3) p=1, q=3

```
CO.arma13 <- arima(e.ts.CO, order=c(1,0,3), include.mean=FALSE)
summary(CO.arma13)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(1, 0, 3), include.mean = FALSE)
##
## Coefficients:
##          ar1      ma1     ma2     ma3
##       0.5889  -0.1777  0.0119  0.1140
## s.e.  0.1797   0.1781  0.0981  0.0773
##
## sigma^2 estimated as 2.294:  log likelihood = -704.45,  aic = 1418.89
##
## Training set error measures:
##                        ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.001465841 1.514552 1.162444 161.5466 268.2974 0.8894476
##                     ACF1
## Training set 0.002203679
```

AIC = 1418.89

arma(1,2) p=1, q=2

```r
CO.arma12 <- arima(e.ts.CO, order=c(1,0,2), include.mean=FALSE)
summary(CO.arma12)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(1, 0, 2), include.mean = FALSE)
##
## Coefficients:
##          ar1      ma1      ma2
##       0.8403  -0.4277  -0.1249
## s.e.  0.0959   0.1149   0.0928
##
## sigma^2 estimated as 2.3:  log likelihood = -704.94,  aic = 1417.89
##
## Training set error measures:
##                       ME   RMSE      MAE      MPE     MAPE      MASE
## Training set -0.00128849 1.5165 1.157077 186.5701 289.4883 0.8853411
##                       ACF1
## Training set -0.0002679074
```

AIC = 1417.89

arma(2,3) p=2, q=3

```r
CO.arma23 <- arima(e.ts.CO, order=c(2,0,3), include.mean=FALSE)
summary(CO.arma23)
```

```
##
## Call:
## arima(x = e.ts.CO, order = c(2, 0, 3), include.mean = FALSE)
##
## Coefficients:
##          ar1      ar2      ma1     ma2     ma3
##       1.5857  -0.5981  -1.1807  0.1826  0.0492
## s.e.  0.1804   0.1725   0.1845  0.1277  0.0764
##
## sigma^2 estimated as 2.283:  log likelihood = -703.57,  aic = 1419.15
##
## Training set error measures:
##                        ME    RMSE     MAE      MPE     MAPE      MASE
## Training set -0.002953586 1.51094 1.15345 174.5896 284.2401 0.8825656
##                      ACF1
## Training set -0.001869394
```

AIC = 1419.15

Based on the above AIC values, we would choose model AR(3) because it has the lowest AIC value. As a final step, we will use the auto.arima function on e.ts.CO.

```r
CO.auto <- auto.arima(e.ts.CO,approximation=FALSE)
summary(CO.auto)
```

```
## Series: e.ts.CO
## ARIMA(3,0,0) with zero mean
##
## Coefficients:
##          ar1     ar2     ar3
##       0.4120  0.0565  0.1344
## s.e.  0.0508  0.0551  0.0510
##
## sigma^2 estimated as 2.308:  log likelihood=-704.08
## AIC=1416.17   AICc=1416.27   BIC=1431.97
##
## Training set error measures:
##                        ME     RMSE     MAE      MPE     MAPE      MASE
## Training set -0.001624096 1.513104 1.15694 177.2785 280.2686 0.8852366
##                     ACF1
## Training set 0.001369499
```

The auto arima function supports the use of AR(3) as our model, with an AIC of 1416.27. We will move forward using AR(3) to model the residuals of our CO.trend.seasonal model.

Modeling e.ts.NO2

Now we will try out some models for e.ts.CO using various p and q values.

ar(1) p=1

```
e.ts.NO2
```

```
## Time Series:
## Start = 1
## End = 384
## Frequency = 1
##            1            2            3            4            5
##   -8.94021914  42.42475108  52.43560271  60.85379355  73.38102627
##            6            7            8            9           10
##   79.41407636  54.75809366  61.25520611  44.62017633  46.63102796
##           11           12           13           14           15
##   33.04921880  42.57645152  36.60950162   7.95351891   1.45063137
##           16           17           18           19           20
##    7.81560159  -0.17354679  25.24464405  -0.22812322  22.80492687
##           21           22           23           24           25
##   33.14894416 -11.35394338  29.01102684  45.02187846  -3.55993070
##           26           27           28           29           30
##   -9.36940461  19.00035212   6.34436941  -9.15851813  34.20645209
##           31           32           33           34           35
##   14.21730371  21.63549455  11.16272728 -21.80422263  14.53979466
##           36           37           38           39           40
##  -20.96309288  19.40187734  22.41272896 -13.16908020 -13.56612959
##           41           42           43           44           45
##  -21.57237470 -36.26723994 -46.80858683 -45.48164225  67.60815422
##           46           47           48           49           50
##   19.02634505  -9.44642222   0.58662787   3.93064517  18.42775762
```

```
##          51          52          53          54          55 
##   3.79272784   4.80357947  14.22177031  10.74900303   9.78205312 
##          56          57          58          59          60 
##  -2.87392958 -20.37681713  -5.01184691   0.99900472 -24.20626142 
##          61          62          63          64          65 
## -19.70440860  12.97747838  26.32149567 -16.18139187   2.18357835 
##          66          67          68          69          70 
##  30.19442997  28.61262081   9.13985354  18.17290363  23.51692092 
##          71          72          73          74          75 
##  16.01403338  31.37900360  19.38985522   3.80804606  -3.66472121 
##          76          77          78          79          80 
## -11.63167112  14.71234617 -12.59582947 -55.48049012 -18.41471953 
##          81          82          83          84          85 
##  -4.99652869   8.53070404  33.56375413 -33.09222858 -57.59511612 
##          86          87          88          89          90 
## -43.23014590 -10.58744799 -14.80110344 -23.27387071  12.75917938 
##          91          92          93          94          95 
##  39.10319667  67.60030913  57.96527935  16.97613098 -14.60567819 
##          96          97          98          99         100 
## -34.07844546 -28.04539537  12.29862193 -14.20426562  -0.83929540 
##         101         102         103         104         105 
##  37.17155623 -16.41025293 -10.88302021  14.15002988   6.49404718 
##         106         107         108         109         110 
##   7.99115963 -28.64387015   6.36698148  12.78517232  10.31240504 
##         111         112         113         114         115 
##  48.34545514  52.68947243  66.18658489  20.55155511  27.56240673 
##         116         117         118         119         120 
##  -2.01940243 -20.49216970  66.54088039  42.88489768  71.38201014 
##         121         122         123         124         125 
##  45.74698036  31.75783198  -7.82397718   5.70325555 -10.26369436 
##         126         127         128         129         130 
##  20.08032293  26.57743539  35.94240561  55.95325723   1.37144807 
##         131         132         133         134         135 
## -18.10131920  59.93173089  38.27574818  28.77286064  26.13783086 
##         136         137         138         139         140 
##   4.14868248 -22.43312668 -23.90589395 -10.87284386 -28.52882657 
##         141         142         143         144         145 
##  -1.03171411   5.33325611  13.34410774  33.76229857 -35.71046870 
##         146         147         148         149         150 
##  29.32258139  26.66659869 -73.41031921 -52.47131864 -35.46046701 
##         151         152         153         154         155 
## -46.04227617 -39.51504345 -49.48199336 -34.13797606 -101.64086361 
##         156         157         158         159         160 
## -83.27589339 -48.26504176 -58.84685092 -48.31961820 -37.28656810 
##         161         162         163         164         165 
## -24.94255081 -75.95847064 -74.47350640 -62.37067093 -48.85878029 
##         166         167         168         169         170 
## -44.23612129 -52.15875429 -34.74712556 -67.25001310 -73.88504288 
##         171         172         173         174         175 
## -36.87419126 -45.45600042 -45.32348773 -53.18082108 -51.41902655 
##         176         177         178         179         180 
##  -5.05458785 -48.68961763  -2.74644222 -41.52389126  35.79181021 
##         181         182         183         184         185 
## -37.94871397 -25.20315153  10.14083740  43.50580762  15.51665924 
```

```
##         186         187         188         189         190 
## -50.06514992 -47.22845854  53.49513290 -69.57728549  -6.66373735 
##         191         192         193         194         195 
## -51.29876713 -37.28791550 -20.86972467  39.65750806  68.69055815 
##         196         197         198         199         200 
##  39.03457545  36.53168790   9.89665812  -8.09249025 -55.67429941 
##         201         202         203         204         205 
## -23.14706669  -0.11401660  36.23000070  20.72711315   6.09208337 
##         206         207         208         209         210 
##  -0.89706500 -54.06712604 -49.36392390 -57.15298429 -71.62916790 
##         211         212         213         214         215 
## -81.95035726 -55.71249138 -43.70163975 -36.28344891 -50.75621618 
##         216         217         218         219         220 
## -52.72316609 -72.12256358 -82.43075380 -80.86927130 -68.66010294 
##         221         222         223         224         225 
## -55.04180143 -50.31267460 -58.07595751 -72.52651082   5.31338891 
##         226         227         228         229         230 
## -67.32164087 -12.31078925 -41.89259841 -29.36536568  -7.33231559 
##         231         232         233         234         235 
##   7.01170170 -58.49118584 -40.12621562 -14.11536400  -3.69717316 
##         236         237         238         239         240 
## -53.16994043 -40.81284872   8.20712695 -68.29576059  -4.93079037 
##         241         242         243         244         245 
##   1.08006126 -24.50174791 -20.97451518  41.04931897 -16.57163871 
##         246         247         248         249         250 
##  -1.10033534  18.26463488  -2.72451349  31.69367734 -56.77908993 
##         251         252         253         254         255 
##  23.25396016  36.59797746  92.09508991  87.46006013  48.47091176 
##         256         257         258         259         260 
##  73.88910260  51.41633532  72.44938542  45.79340271 -16.70948484 
##         261         262         263         264         265 
##  -6.34451462  70.66633701  16.08452785 -34.38823943  57.64481067 
##         266         267         268         269         270 
## -11.01117204 -14.51405958 -27.14908936 -13.13823774 -48.72004690 
##         271         272         273         274         275 
## -62.19281417 -15.15976408 -35.81574679 -74.31863433 -64.95366411 
##         276         277         278         279         280 
## -10.94281249 -27.52462165  18.00261108  91.03566117  64.37967846 
##         281         282         283         284         285 
##  25.87679092  34.24176114  23.25261276   2.67080360   3.19803633 
##         286         287         288         289         290 
## -28.76891358   3.57510371 -26.92778383  48.43718639  25.44803802 
##         291         292         293         294         295 
##  12.82676459 -27.60653842 -43.57348833 -72.43969899 -82.63890066 
##         296         297         298         299         300 
## -80.96923918 -68.65270141  11.06165410  -1.41111317   6.62193692 
##         301         302         303         304         305 
##  -2.03404578  10.46306667 -38.17196311 -15.16111148  14.25707936 
##         306         307         308         309         310 
## -12.21568792 -30.18263782 -37.83862053 -14.34150808 -30.97653786 
##         311         312         313         314         315 
##   0.03431377  28.45250461 -12.02026267  85.01278743  41.35680472 
##         316         317         318         319         320 
## -60.14608282  -0.78111260  45.22973902  60.64792986  25.17516259 
```

```
##           321          322          323          324          325
##   14.20821268 -15.44777003 -28.95065757  -8.58568735   5.42516427
##           326          327          328          329          330
##  -47.15664489 -18.62941216  94.40363793  21.74765522  14.24476768
##           331          332          333          334          335
##  118.60973790 103.62058952 121.03878036  27.56601309  88.59906318
##           336          337          338          339          340
##   45.94308047  27.44019293  72.80516315 135.81601478  74.23420561
##           341          342          343          344          345
##   42.76143834  17.79448843  11.13850572  68.63561818  20.00058840
##           346          347          348          349          350
##   44.01144003  57.42963086 -14.04313641  86.98991368  84.33393098
##           351          352          353          354          355
##   27.83104343  50.19601365  17.92236887 -25.37494388  -0.84771116
##           356          357          358          359          360
##  -47.81466106 -53.47064377  17.02646868   6.39143890  15.40229053
##           361          362          363          364          365
##    2.82048137   2.34771409  24.38076419  31.72478148  10.22189394
##           366          367          368          369          370
##   21.58686416  23.59771578  23.01590662  21.54313935  32.57618944
##           371          372          373          374          375
##   31.92020673   2.41731919  14.78228941  27.79314103   0.21133187
##           376          377          378          379          380
##  -16.26143540  28.77161469 -19.88436802   1.61274444   4.97771466
##           381          382          383          384
##  -19.01143372 -73.59324288 -71.06601015 -30.03296006
```

```r
NO2.ar1 <- arima(e.ts.NO2, order=c(1,0,0), include.mean=FALSE)
summary(NO2.ar1)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(1, 0, 0), include.mean = FALSE)
##
## Coefficients:
##          ar1
##       0.6165
## s.e.  0.0400
##
## sigma^2 estimated as 1053:  log likelihood = -1881.37,  aic = 3766.75
##
## Training set error measures:
##                       ME     RMSE      MAE      MPE     MAPE     MASE
## Training set -0.04326599 32.45481 24.84612 151.2414 370.8021 0.909382
##                    ACF1
## Training set -0.1031
```

AIC = 3766.75

ar(2) p=2

```r
NO2.ar2 <- arima(e.ts.NO2, order=c(2,0,0), include.mean=FALSE)
summary(NO2.ar2)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(2, 0, 0), include.mean = FALSE)
##
## Coefficients:
##          ar1     ar2
##       0.5110  0.1709
## s.e.  0.0502  0.0505
##
## sigma^2 estimated as 1023:  log likelihood = -1875.73,  aic = 3757.46
##
## Training set error measures:
##                        ME     RMSE    MAE    MPE     MAPE      MASE
## Training set -0.07913584 31.97892 24.617 132.05 369.6772 0.9009962
##                     ACF1
## Training set -0.02892163
```

AIC = 3757.73

ar(3) p=3

```r
NO2.ar3 <- arima(e.ts.NO2, order=c(3,0,0), include.mean=FALSE)
summary(NO2.ar3)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(3, 0, 0), include.mean = FALSE)
##
## Coefficients:
##          ar1     ar2     ar3
##       0.4793  0.0792  0.1820
## s.e.  0.0502  0.0558  0.0505
##
## sigma^2 estimated as 988.9:  log likelihood = -1869.33,  aic = 3746.67
##
## Training set error measures:
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.1390427 31.44647 24.10174 174.7213 398.0462 0.8821373
##                     ACF1
## Training set 0.003605925
```

AIC = 3746.67

ma(1) p=0, q=1

```r
NO2.ma1 <- arima(e.ts.NO2, order=c(0,0,1), include.mean=FALSE)
summary(NO2.ma1)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(0, 0, 1), include.mean = FALSE)
##
## Coefficients:
```

```
##          ma1
##       0.4935
## s.e.  0.0387
##
## sigma^2 estimated as 1232:  log likelihood = -1911.4,  aic = 3826.81
##
## Training set error measures:
##                        ME     RMSE      MAE      MPE     MAPE    MASE
## Training set -0.009648994 35.10391 27.46488 176.0653 285.5227 1.00523
##                   ACF1
## Training set 0.1400909
```

AIC = 3826.81

ma(2) p=0, q=2

```r
NO2.ma2 <- arima(e.ts.NO2, order=c(0,0,2), include.mean=FALSE)
summary(NO2.ma2)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(0, 0, 2), include.mean = FALSE)
##
## Coefficients:
##          ma1     ma2
##       0.5267  0.2153
## s.e.  0.0504  0.0432
##
## sigma^2 estimated as 1159:  log likelihood = -1899.61,  aic = 3805.23
##
## Training set error measures:
##                        ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.008959196 34.04144 26.45011 140.3266 306.4735 0.9680891
##                    ACF1
## Training set 0.05900395
```

AIC = 3805.23

ma(3) p=0, q=3

```r
NO2.ma3 <- arima(e.ts.NO2, order=c(0,0,3), include.mean=FALSE)
summary(NO2.ma3)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(0, 0, 3), include.mean = FALSE)
##
## Coefficients:
##          ma1     ma2    ma3
##       0.4718  0.2886  0.296
## s.e.  0.0510  0.0456  0.048
##
## sigma^2 estimated as 1067:  log likelihood = -1883.82,  aic = 3775.63
```

```
##
## Training set error measures:
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.03594799 32.66221 25.38279 121.5382 293.1753 0.9290244
##                    ACF1
## Training set 0.06087396
```

AIC = 3775.63

arma(1,2) p=1, q=2

```r
NO2.arma12 <- arima(e.ts.NO2, order=c(1,0,2), include.mean=FALSE)
summary(NO2.arma12)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(1, 0, 2), include.mean = FALSE)
##
## Coefficients:
##          ar1      ma1      ma2
##       0.9054  -0.4284  -0.1351
## s.e.  0.0395   0.0669   0.0675
##
## sigma^2 estimated as 992.1:  log likelihood = -1869.97,  aic = 3747.93
##
## Training set error measures:
##                      ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.1802244 31.49776 24.19366 183.6265 390.1247 0.8855017
##                      ACF1
## Training set -0.001293883
```

AIC = 3747.93

arma(1,3) p=1, q=3

```r
NO2.arma13 <- arima(e.ts.NO2, order=c(1,0,3), include.mean=FALSE)
summary(NO2.arma13)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(1, 0, 3), include.mean = FALSE)
##
## Coefficients:
##          ar1      ma1      ma2     ma3
##       0.8845  -0.4047  -0.1342  0.0399
## s.e.  0.0619   0.0856   0.0732  0.0609
##
## sigma^2 estimated as 990.9:  log likelihood = -1869.74,  aic = 3749.47
##
## Training set error measures:
##                      ME     RMSE      MAE      MPE    MAPE      MASE
## Training set -0.1671569 31.47917 24.14297 184.5395 391.305 0.8836462
##                       ACF1
## Training set -0.0007001616
```

AIC = 3749.47

arma(2,3) p=2, q=3

```
NO2.arma23 <- arima(e.ts.NO2, order=c(2,0,3), include.mean=FALSE)
summary(NO2.arma23)
```

```
##
## Call:
## arima(x = e.ts.NO2, order = c(2, 0, 3), include.mean = FALSE)
##
## Coefficients:
##          ar1      ar2      ma1     ma2     ma3
##       1.5433  -0.5732  -1.0670  0.1440  0.0668
## s.e.  0.4899   0.4456   0.4906  0.2284  0.0906
##
## sigma^2 estimated as 991.6:  log likelihood = -1869.87,  aic = 3751.75
##
## Training set error measures:
##                      ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.1859082 31.49008 24.17697 182.4941 392.9221 0.8848909
##                      ACF1
## Training set -0.002285031
```

AIC = 3751.75

Based on the above AIC values, we would choose model AR(3) because it has the lowest AIC value. As a final step, we will use the auto.arima function on e.ts.NO2.

```
NO2.auto <- auto.arima(e.ts.NO2,approximation=FALSE)
summary(NO2.auto)
```

```
## Series: e.ts.NO2
## ARIMA(1,1,2)
##
## Coefficients:
##          ar1      ma1     ma2
##       0.8271  -1.3734  0.3832
## s.e.  0.0810   0.1150  0.1037
##
## sigma^2 estimated as 1018:  log likelihood=-1869.01
## AIC=3746.02   AICc=3746.13   BIC=3761.82
##
## Training set error measures:
##                      ME    RMSE      MAE      MPE    MAPE      MASE
## Training set -0.6548743 31.7411 24.27322 171.3796 441.267 0.8884137
##                    ACF1
## Training set 0.02459958
```

AIC = 3746.13 Because of the observations from the PACF and its low AIC, we will move forward using AR(3) for the residuals for our NO2.trend.seasonal model.

Part D: Assessment of Models – update these

We used AIC and diagnostics to assess the models for CO.

```r
AIC(CO.ar1)
```

```
## [1] 1423.979
```

```r
AIC(CO.ar2)
```

```
## [1] 1421.049
```

```r
AIC(CO.ar3)
```

```
## [1] 1416.168
```

```r
AIC(CO.ma1)
```

```
## [1] 1449.388
```

```r
AIC(CO.ma2)
```

```
## [1] 1438.973
```

```r
AIC(CO.ma3)
```

```
## [1] 1426.24
```

```r
AIC(CO.arma12)
```

```
## [1] 1417.886
```

```r
AIC(CO.arma13)
```

```
## [1] 1418.894
```

```r
AIC(CO.arma23)
```

```
## [1] 1419.147
```

```r
AIC(CO.auto)
```

```
## [1] 1416.168
```

The lowest AIC is the CO.ar3, which is what the auto.arima function produced as well. Therefore the model we would choose is AR(3).

We also used AIC and diagnostics to assess the models for N02.

```r
AIC(NO2.ar1)
```

```
## [1] 3766.747
```

```r
AIC(NO2.ar2)
```

```
## [1] 3757.461
```

```r
AIC(NO2.ar3)
```

```
## [1] 3746.669
```

```r
AIC(NO2.ma1)
```

```
## [1] 3826.808
```

```r
AIC(NO2.ma2)
```

```
## [1] 3805.228
```

```r
AIC(NO2.ma3)
```

```
## [1] 3775.632
```

```r
AIC(NO2.arma12)
```

```
## [1] 3747.932
```

```r
AIC(NO2.arma13)
```

```
## [1] 3749.472
```

```r
AIC(NO2.arma23)
```

```
## [1] 3751.746
```

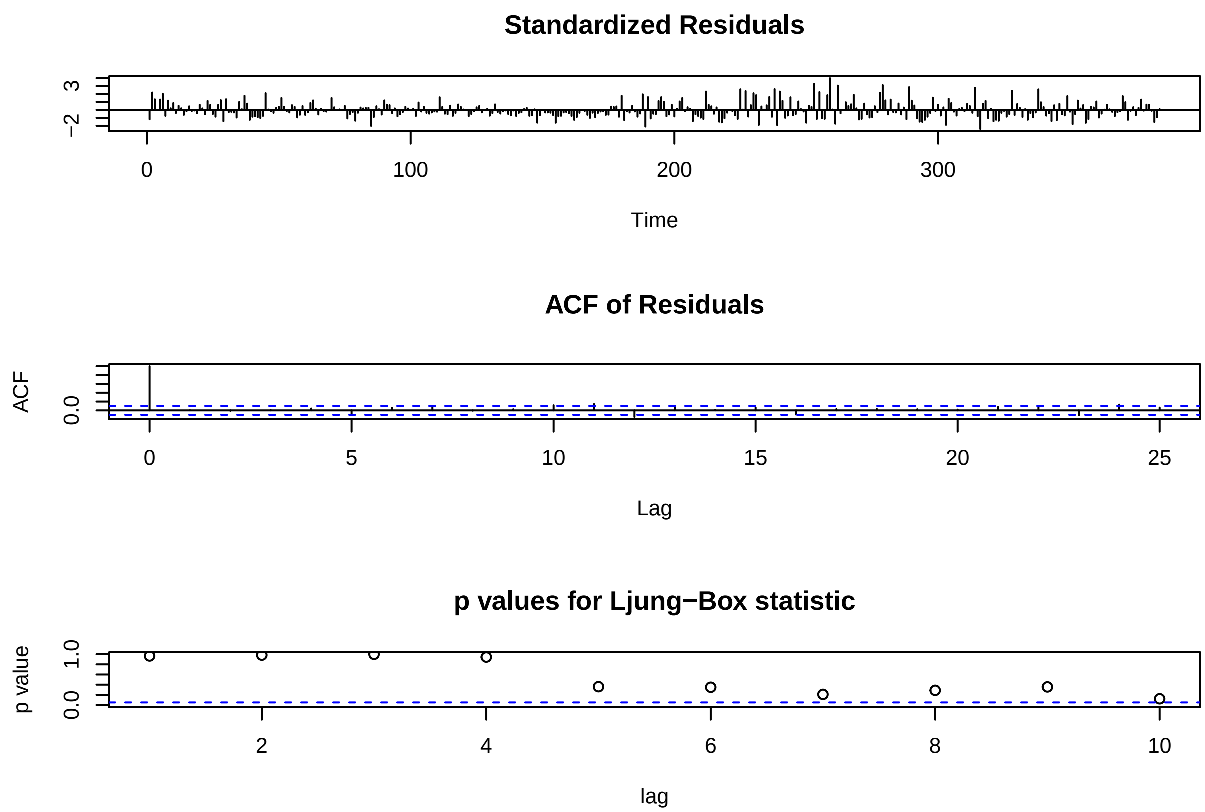
```r
AIC(NO2.auto)
```

```
## [1] 3746.025
```

The lowest AIC is the NO2.ar(3), therefore we will use this model.

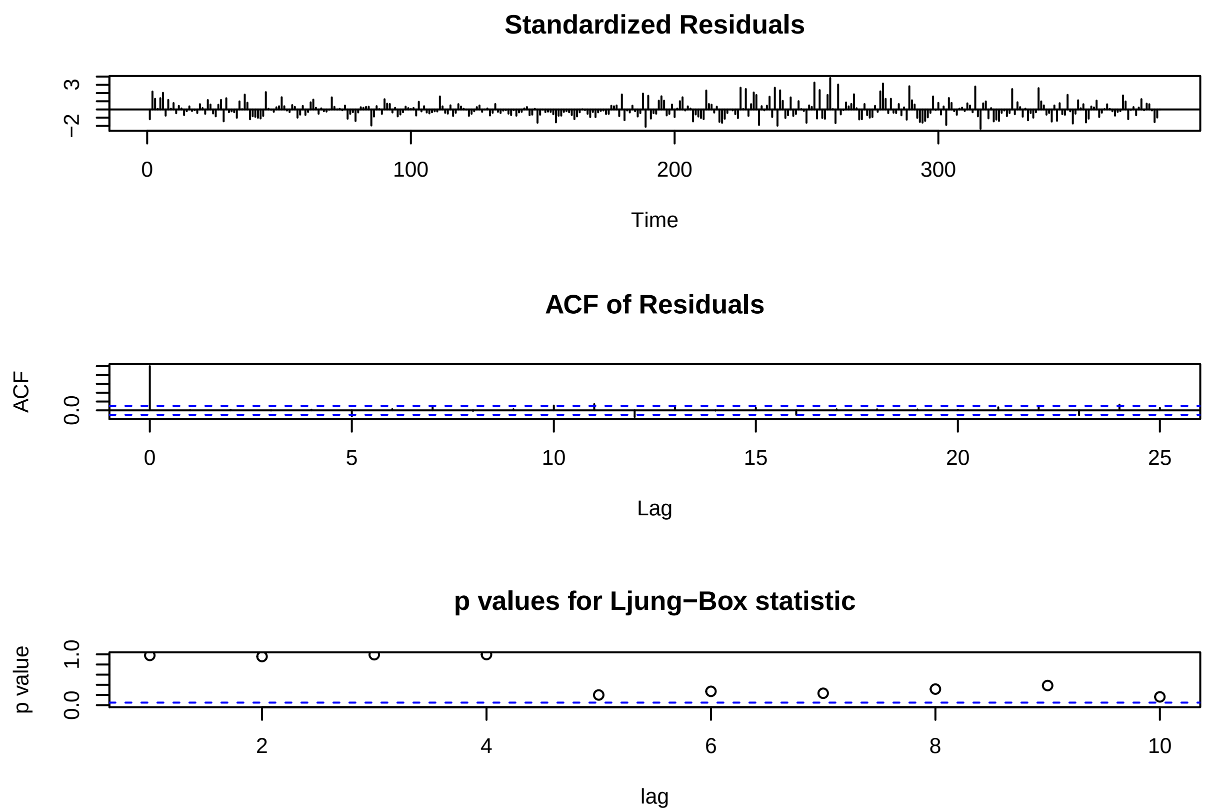
Now we will consider diagnostics of the best CO model and second best CO model according to their AIC.

```r
tsdiag(CO.arma13, lag = 30)
```



The standardized residuals graph shows that there is noise in the ARMA(1,3) model of the residuals. The ACF of residuals shows no significant lags in the ARMA(1,3) model. Finally, the Ljung-Box statistic shows that at lag 10, the pvalue is very, very close to 0. While it is still adequate for lags 1-9, we could do better.

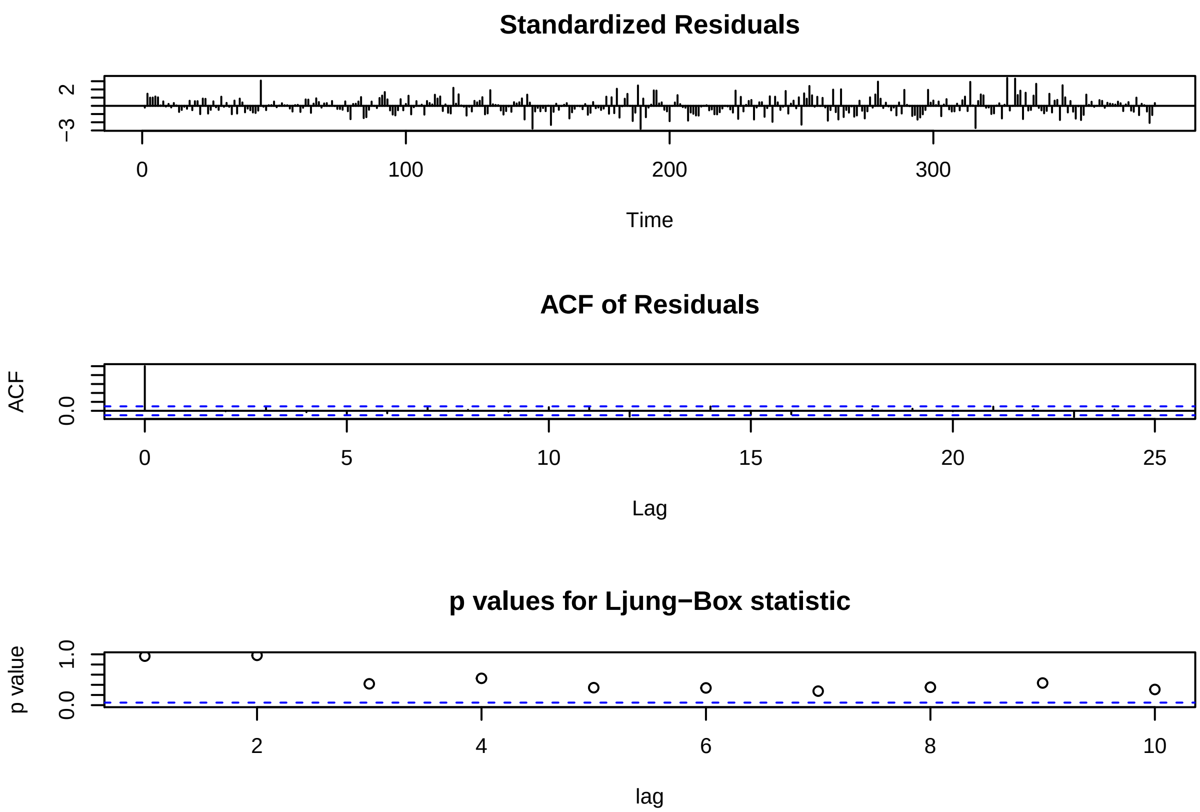
```
tsdiag(CO.ar3, lag = 30)
```



The standardized residuals graph shows that there is noise in the AR(3) model of the residuals. The ACF of residuals shows no significant lags in the AR(3) model. Finally, the Ljung-Box statistic shows that the p-values do not touch 0, and is at adequate at all the lags. All of these attributes from the diagnostics plots indicate a valid model.

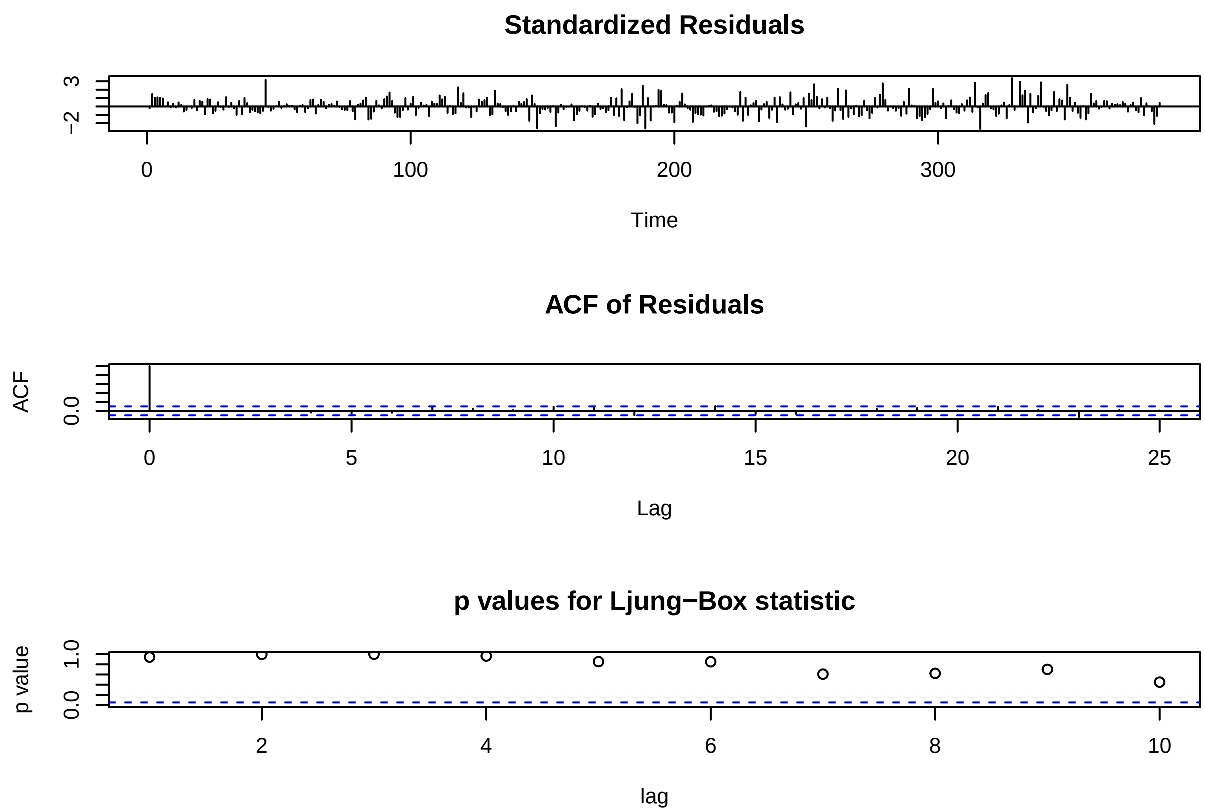
Now we will consider diagnostics of the best NO2 model and second best NO2 model according to their AIC.

```
tsdiag(NO2.arma23, lag = 30)
```



The standardized residuals graph shows that there is noise in the ARMA(2,3) model of the residuals. The ACF of residuals shows no significant lags in the ARMA(2,3) model. Finally, the Ljung-Box statistic shows that the p-values do not touch 0, and is at adequate at all the lags. All of these attributes from the diagnotics plots indicate a valid model. Now, let's look at the model with the best AIC, AR(3).

```
tsdiag(NO2.ar3, lag = 30)
```



The standardized residuals graph shows that there is noise in the AR(3) model of the residuals. The ACF of residuals shows no significant lags in the AR(3) model. Finally, the Ljung-Box statistic shows that the p-values do not touch 0, and is at adequate at all the lags. All of these attributes from the diagnostics plots indicate a valid model. Because this model also had the lowest AIC, we will use it moving forward.

Part E: Diagnostics

Regarding the diagnostics of the models we have chosen, there does not appear to be any concern that would warrant a transformation, such as a log transform. The diagnostics also do not demonstrate that our models violate assumptions, so we can conclude that they are valid models to use.

Part 2: Building Multivariate Time Series Models

Part A: Seasonality

We used the same models as we used in Part 1 because we found that these were valid models that did not violate assumptions. The models in Part 1 demonstrated seasonality, which we were able to observe from the periodograms that the models produced. These periodograms showed peaks at a frequency around 0.14. This would make the period 1/0.14, or about 7. This makes intuitive sense because this would be a weekly period. Thus, we added a seasonality component to the models to account for the seasonality of the models. So, since we will be using the same models from part 1, we do not need to execute this process again.

Part B: Trends

Again, since we are using the same models as we used in Part 1, we can conclude that the data for both CO and NO2 show trends, so we will need to account for trend when building our model. In part 1 we demonstrated this by building linear models to model the trend component. The models showed significant p-values indicating a trend in the data. So, our models accounted for trend. Since we are using the same models, this analysis still applies here for the multivariate time series model.

Part C: Auto-Regressive and Moving Average

```
allResiduals <- data.frame(e.ts.CO, e.ts.NO2)
colnames(allResiduals) <- c("CO","NO2")
cor(allResiduals)
```

```
##            CO       NO2
## CO  1.0000000 0.5891419
## NO2 0.5891419 1.0000000
```

Correlation between residuals of NO2 and CO is 0.58914.

Build VARMA model to CO and NO2 residuals

Because our univariate models both had auto-regressive and moving average components, we can create a varma model that will evaluate different p and q values, which we can interpret to determine which model is best.

Part D: Assessment of Models

We will analyze the AICmatrix to find which model has the lowest AIC.

```
AICmatrix
```

```
##          [,1]      [,2]     [,3]     [,4]
## [1,] 7.245625  7.493295 7.669472 7.541659
## [2,] 7.406789 10.508200 7.163330 8.063202
## [3,] 7.484692  9.111086 7.466969 7.188300
```

According to the AICmatrix, the model with p=2 and q=2 has the lowest AIC, this we should use these values to build our model. The AIC for p=2 and q=2 is 7.16333. The next best model is p=3 and q=3, with an AIC of 7.1883 . We will build these 2 models and compare them using diagnostics.

The CCF plots shows the correlations are all within the dashed line, meaning that the correlations are not statistically different from 0. The significance plot of CCM shows a few lags that are below the dashed line. While this is not too problematic, we can do better. The plot of p-values for Ljung-Box statistics shows that the model is adequate until around lag 12. The residual plot for CO and NO2 show noise, *yay!*

```
varma.model <- VARMACpp(allResiduals, p=2, q=2, include.mean=F)
```

```
## Number of parameters:  16
## initial estimates:  0.7605 -0.0125 -0.0601 0.0092 5.8783 0.5354 -6.8113 0.3256 -0.3465 0.012 -0.0364
## Par. lower-bounds:  0.2433 -0.0368 -0.4857 -0.0125 -4.9125 0.0292 -15.6904 -0.1279 -0.8821 -0.0131 -0
## Par. upper-bounds:  1.2777 0.0117 0.3654 0.031 16.669 1.0415 2.0678 0.779 0.1892 0.0371 0.2411 0.0114
## Final   Estimates:  0.7604869 -0.01254823 -0.06011082 0.009244218 5.878253 0.5353698 -6.811301 0.3255
##
## Coefficient(s):
##      Estimate  Std. Error  t value Pr(>|t|)
## CO   0.760487          NA       NA       NA
## NO2 -0.012548          NA       NA       NA
## CO  -0.060111          NA       NA       NA
## NO2  0.009244          NA       NA       NA
## CO   5.878253          NA       NA       NA
## NO2  0.535370          NA       NA       NA
## CO  -6.811301          NA       NA       NA
## NO2  0.325555          NA       NA       NA
##     -0.346455          NA       NA       NA
##      0.012014          NA       NA       NA
##     -0.036432          NA       NA       NA
##     -0.001438          NA       NA       NA
##     -6.906632          NA       NA       NA
##     -0.061093          NA       NA       NA
##      2.676609          NA       NA       NA
##     -0.223547          NA       NA       NA
## ---
## Estimates in matrix form:
## AR coefficient matrix
## AR( 1 )-matrix
##      [,1]    [,2]
## [1,] 0.76 -0.0125
## [2,] 5.88  0.5354
## AR( 2 )-matrix
##         [,1]    [,2]
## [1,] -0.0601 0.00924
## [2,] -6.8113 0.32556
## MA coefficient matrix
## MA( 1 )-matrix
##       [,1]    [,2]
## [1,] 0.346 -0.0120
## [2,] 6.907  0.0611
## MA( 2 )-matrix
##         [,1]    [,2]
## [1,]  0.0364 0.00144
## [2,] -2.6766 0.22355
##
## Residuals cov-matrix:
##          [,1]      [,2]
## [1,]  2.27996  32.42389
## [2,] 32.42389 982.15475
## ----
## aic=  7.16333
## bic=  7.32794
```

```
varma.model
```

```
## $data
##              CO           NO2
## 1   -2.090038263   -8.94021914
## 2    2.343716565   42.42475108
## 3    2.744075476   52.43560271
## 4    1.081433762   60.85379355
## 5    2.998325429   73.38102627
## 6    4.729206463   79.41407636
## 7    1.125112849   54.75809366
## 8    2.889323315   61.25520611
## 9    2.023078142   44.62017633
## 10   2.323437053   46.63102796
## 11   0.760795339   33.04921880
## 12   1.377687006   42.57645152
## 13   1.108568040   36.60950162
## 14  -0.395525573    7.95351891
## 15  -0.231315108    1.45063137
## 16   0.602439719    7.81560159
## 17  -0.097201369   -0.17354679
## 18  -0.159843084   25.24464405
## 19  -0.642951417   -0.22812322
## 20   0.687929617   22.80492687
## 21   0.483836004   33.14894416
## 22  -0.651953531  -11.35394338
## 23   1.581801296   29.01102684
## 24   1.582160208   45.02187846
## 25  -0.080481506   -3.55993070
## 26  -0.999987702   -9.36940461
## 27   0.667291195   19.00035212
## 28   1.963197581    6.34436941
## 29  -1.472591954   -9.15851813
## 30   1.661162873   34.20645209
## 31   0.361521785   14.21730371
## 32  -0.301119929   21.63549455
## 33  -0.384228262   11.16272728
## 34  -1.653347228  -21.80422263
## 35   0.742559158   14.53979466
## 36   0.206769623  -20.96309288
## 37   2.640524451   19.40187734
## 38   2.440883362   22.41272896
## 39  -0.621758352  -13.16908020
## 40  -1.015520814  -13.56612959
## 41  -1.482819267  -21.57237470
## 42  -2.285095184  -36.26723994
## 43  -2.819069910  -46.80858683
## 44  -2.683503231  -45.48164225
## 45   1.620244939   67.60815422
## 46   0.257603225   19.02634505
## 47  -0.225505108   -9.44642222
## 48  -0.294624074    0.58662787
## 49   0.301282312    3.93064517
## 50   0.665492778   18.42775762
```

```
## 51   2.499247605    3.79272784
## 52   1.699606516    4.80357947
## 53   0.736964802   14.22177031
## 54   0.253856469   10.74900303
## 55   1.184737503    9.78205312
## 56   1.080643890   -2.87392958
## 57  -0.955145645  -20.37681713
## 58  -1.121390818   -5.01184691
## 59   0.278968094    0.99900472
## 60  -1.083673621  -24.20626142
## 61  -1.039890068  -19.70440860
## 62   0.864099080   12.97747838
## 63   1.960005467   26.32149567
## 64   1.024215932  -16.18139187
## 65  -0.142029241    2.18357835
## 66   0.458329671   30.19442997
## 67  -0.004312043   28.61262081
## 68  -0.387420377    9.13985354
## 69  -0.156539342   18.17290363
## 70   2.139367044   23.51692092
## 71   1.303577509   16.01403338
## 72   0.637332337   31.37900360
## 73   0.737691248   19.38985522
## 74   0.375049534    3.80804606
## 75   0.991941201   -3.66472121
## 76  -1.177177765  -11.63167112
## 77  -1.181271379   14.71234617
## 78  -0.917060914  -12.59582947
## 79  -2.725687913  -55.48049012
## 80  -1.882947175  -18.41471953
## 81  -0.645588889   -4.99652869
## 82  -0.428697222    8.53070404
## 83   0.002183812   33.56375413
## 84   0.398090198  -33.09222858
## 85  -2.847354773  -57.59511612
## 86  -2.450425326  -43.23014590
## 87  -0.503585598  -10.58744799
## 88  -0.666227312  -14.80110344
## 89  -1.449335645  -23.27387071
## 90   1.181545389   12.75917938
## 91   1.377451775   39.10319667
## 92   1.441662241   67.60030913
## 93   0.275417068   57.96527935
## 94   0.675775979   16.97613098
## 95  -0.786865735  -14.60567819
## 96  -1.169974068  -34.07844546
## 97  -0.939093034  -28.04539537
## 98  -0.043186647   12.29862193
## 99   0.021023818  -14.20426562
## 100 -0.045221355   -0.83929540
## 101  0.255137557   37.17155623
## 102 -1.007504158  -16.41025293
## 103  1.009387509  -10.88302021
## 104  0.040268543   14.15002988
```

```
## 105  0.536174930    6.49404718
## 106 -0.199614605    7.99115963
## 107 -0.765859778  -28.64387015
## 108 -0.765500866    6.36698148
## 109 -0.728142580   12.78517232
## 110 -0.811250914   10.31240504
## 111  1.919630121   48.34545514
## 112  1.215536507   52.68947243
## 113 -0.120253028   66.18658489
## 114 -0.486498200   20.55155511
## 115  0.713860711   27.56240673
## 116 -0.948781003   -2.01940243
## 117 -1.031889336  -20.49216970
## 118  0.598991698   66.54088039
## 119  0.594898084   42.88489768
## 120  0.259108549   71.38201014
## 121  0.192863377   45.74698036
## 122 -1.006777712   31.75783198
## 123 -1.169419426   -7.82397718
## 124 -0.852527759    5.70325555
## 125 -0.121646725  -10.26369436
## 126  0.474259661   20.08032293
## 127 -0.361529873   26.57743539
## 128 -0.127775046   35.94240561
## 129  0.172583865   55.95325723
## 130 -1.090057849    1.37144807
## 131 -1.073166182  -18.10131920
## 132  0.520627305   59.93173089
## 133 -0.424380775   38.27574818
## 134 -0.939638569   28.77286064
## 135 -0.548413469   26.13783086
## 136 -0.448054557    4.14868248
## 137 -1.110696272  -22.43312668
## 138 -1.593804605  -23.90589395
## 139 -0.889685393  -10.87284386
## 140 -1.754167757  -28.52882657
## 141 -1.523485425   -1.03171411
## 142 -1.288853805    5.33325611
## 143 -0.669151197   13.34410774
## 144 -0.113315596   33.76229857
## 145 -1.314443028  -35.71046870
## 146 -1.583561994   29.32258139
## 147 -0.625415495   26.66659869
## 148 -2.968063964  -73.41031921
## 149 -2.439655771  -52.47131864
## 150 -1.289331403  -35.46046701
## 151 -1.551973117  -46.04227617
## 152 -1.435081451  -39.51504345
## 153 -1.304200416  -49.48199336
## 154 -1.708294030  -34.13797606
## 155 -3.344083565 -101.64086361
## 156 -2.810328737  -83.27589339
## 157 -2.709969826  -48.26504176
## 158 -2.172611540  -58.84685092
```

```
## 159 -1.755719873  -48.31961820
## 160 -1.724838839  -37.28656810
## 161 -2.128932453  -24.94255081
## 162 -2.995073149  -75.95847064
## 163 -2.860036185  -74.47350640
## 164 -2.158728920  -62.37067093
## 165 -1.420433716  -48.85878029
## 166 -1.202616634  -44.23612129
## 167 -1.671315316  -52.15875429
## 168 -2.349570876  -34.74712556
## 169 -1.685360410  -67.25001310
## 170 -2.451605583  -73.88504288
## 171 -1.951246672  -36.87419126
## 172 -1.434839758  -45.45600042
## 173 -1.217105170  -45.32348773
## 174 -1.685393280  -53.18082108
## 175 -1.770209299  -51.41902655
## 176 -0.305998833   -5.05458785
## 177  0.127755994  -48.68961763
## 178  0.528114906   -2.74644222
## 179 -1.034526809  -41.52389126
## 180  2.382364858   35.79181021
## 181 -0.986754108  -37.94871397
## 182 -0.590847721  -25.20315153
## 183 -0.526637256   10.14083740
## 184  0.307117571   43.50580762
## 185 -0.292523517   15.51665924
## 186 -1.465439134  -50.06514992
## 187 -1.247879833  -47.22845854
## 188  2.292607470   53.49513290
## 189 -2.519796351  -69.57728549
## 190  1.452724321   -6.66373735
## 191 -0.913520852  -51.29876713
## 192 -1.313161940  -37.28791550
## 193 -1.175803654  -20.86972467
## 194  0.941088013   39.65750806
## 195  2.571969047   68.69055815
## 196  2.567875433   39.03457545
## 197  0.232085898   36.53168790
## 198 -0.234159274    9.89665812
## 199  1.166199637   -8.09249025
## 200 -0.896442077  -55.67429941
## 201 -0.179550410  -23.14706669
## 202  1.551330624   -0.11401660
## 203  2.747237010   36.23000070
## 204  1.011447475   20.72711315
## 205  1.345202303    6.09208337
## 206  1.145561214   -0.89706500
## 207 -1.516133271  -54.06712604
## 208 -1.298852898  -49.36392390
## 209 -1.767595954  -57.15298429
## 210 -2.571326381  -71.62916790
## 211 -3.106765385  -81.95035726
## 212  1.824563880  -55.71249138
```

```
## 213  1.224922791  -43.70163975
## 214  1.062281077  -36.28344891
## 215  0.179172744  -50.75621618
## 216  0.810053778  -52.72316609
## 217 -1.794039835  -72.12256358
## 218 -3.125298536  -82.43075380
## 219 -2.991291893  -80.86927130
## 220 -2.290693190  -68.66010294
## 221 -1.553107067  -55.04180143
## 222 -1.335999447  -50.31267460
## 223 -1.804914274  -58.07595751
## 224 -2.608815492  -72.52651082
## 225  2.649532207    5.31338891
## 226  0.783287034  -67.32164087
## 227  3.883645946  -12.31078925
## 228  0.821004232  -41.89259841
## 229  1.637895898  -29.36536568
## 230  4.368776933   -7.33231559
## 231  4.664683319    7.01170170
## 232 -0.471106216  -58.49118584
## 233  1.262648611  -40.12621562
## 234  0.963007523  -14.11536400
## 235  1.100365809   -3.69717316
## 236  3.017257476  -53.16994043
## 237  0.048138510  -40.81284872
## 238  4.344044896    8.20712695
## 239 -0.791744639  -68.29576059
## 240  3.442010189   -4.93079037
## 241  3.542369100    1.08006126
## 242 -0.020272614  -24.50174791
## 243 -0.403380947  -20.97451518
## 244  2.527500087   41.04931897
## 245 -0.176593527  -16.57163871
## 246 -0.812383061   -1.10033534
## 247  1.521371766   18.26463488
## 248  0.621730677   -2.72451349
## 249 -0.040911037   31.69367734
## 250 -2.224019370  -56.77908993
## 251 -0.093138336   23.25396016
## 252  0.302768050   36.59797746
## 253  4.766978516   92.09508991
## 254  0.300733343   87.46006013
## 255  4.001092254   48.47091176
## 256  0.738450540   73.88910260
## 257 -1.144657793   51.41633532
## 258  2.786223241   72.44938542
## 259  6.982129628   45.79340271
## 260  2.946340093  -16.70948484
## 261 -0.519905080   -6.34451462
## 262  5.480453832   70.66633701
## 263  1.717812117   16.08452785
## 264  0.834703784  -34.38823943
## 265  2.465584818   57.64481067
## 266  1.861491205  -11.01117204
```

```
## 267  2.025701670  -14.51405958
## 268  4.059456497  -27.14908936
## 269  2.259815409  -13.13823774
## 270 -0.402826305  -48.72004690
## 271 -1.285934639  -62.19281417
## 272  0.744946396  -15.15976408
## 273 -0.859147218  -35.81574679
## 274 -1.994936753  -74.31863433
## 275 -2.161181925  -64.95366411
## 276 -0.460823014  -10.94281249
## 277 -1.023464728  -27.52462165
## 278  2.593426939   18.00261108
## 279  5.724307973   91.03566117
## 280  4.320214359   64.37967846
## 281  1.784424824   25.87679092
## 282  3.718179652   34.24176114
## 283  1.618538563   23.25261276
## 284  0.455896849    2.67080360
## 285  1.772788516    3.19803633
## 286 -0.096330450  -28.76891358
## 287  0.499575936    3.57510371
## 288 -1.436213598  -26.92778383
## 289  3.697541229   48.43718639
## 290  3.197900140   25.44803802
## 291  2.235258426   12.82676459
## 292  0.052150093  -27.60653842
## 293 -1.716968873  -43.57348833
## 294 -2.826319406  -72.43969899
## 295 -3.362545944  -82.63890066
## 296 -3.229253746  -80.96923918
## 297 -2.529364160  -68.65270141
## 298  0.714620003   11.06165410
## 299 -0.368488330   -1.41111317
## 300  0.762392704    6.62193692
## 301 -0.541700909   -2.03404578
## 302  0.322509556   10.46306667
## 303 -2.643735617  -38.17196311
## 304  0.956623295  -15.16111148
## 305  1.493981580   14.25707936
## 306  0.010873247  -12.21568792
## 307 -0.758245719  -30.18263782
## 308  0.037660668  -37.83862053
## 309  0.301871133  -14.34150808
## 310 -0.264374040  -30.97653786
## 311  1.035984872    0.03431377
## 312  1.173343158   28.45250461
## 313 -0.009765176  -12.02026267
## 314  4.421115859   85.01278743
## 315  0.717022245   41.35680472
## 316 -3.018767290  -60.14608282
## 317  0.514987538   -0.78111260
## 318  1.615346449   45.22973902
## 319 -1.347295265   60.64792986
## 320 -0.154053058   25.17516259
```

```
## 321 -2.119315562   14.20821268
## 322 -2.924023761  -15.44777003
## 323 -3.460432747  -28.95065757
## 324 -2.505650885   -8.58568735
## 325 -1.805291974    5.42516427
## 326 -2.567933688  -47.15664489
## 327 -2.151042021  -18.62941216
## 328  2.479839013   94.40363793
## 329 -0.324254601   21.74765522
## 330  1.039955865   14.24476768
## 331  1.173710692  118.60973790
## 332 -0.825930397  103.62058952
## 333  0.011427889  121.03878036
## 334 -1.871680444   27.56601309
## 335 -1.540799410   88.59906318
## 336 -2.244893024   45.94308047
## 337 -1.780682558   27.44019293
## 338  2.853072269   72.80516315
## 339  2.253431181  135.81601478
## 340  1.590789466   74.23420561
## 341  0.207681133   42.76143834
## 342 -0.161437833   17.79448843
## 343 -2.065531446   11.13850572
## 344 -0.101320981   68.63561818
## 345 -2.267566154   20.00058840
## 346 -0.067207242   44.01144003
## 347 -1.029848956   57.42963086
## 348 -1.712957290  -14.04313641
## 349  1.917923744   86.98991368
## 350  0.213830131   84.33393098
## 351 -2.621959404   27.83104343
## 352 -1.588204577   50.19601365
## 353  0.912154335   17.92236887
## 354  0.149512621  -25.37494388
## 355  0.866404287   -0.84771116
## 356 -1.902714678  -47.81466106
## 357 -2.406808292  -53.47064377
## 358 -0.442597827   17.02646868
## 359 -0.208842999    6.39143890
## 360  1.191515912   15.40229053
## 361 -0.971125802    2.82048137
## 362 -0.954234135    2.34771409
## 363 -0.223353101   24.38076419
## 364  0.672553285   31.72478148
## 365  0.236763750   10.22189394
## 366 -0.229481422   21.58686416
## 367 -1.129122511   23.59771578
## 368 -0.891764225   23.01590662
## 369 -0.774872558   21.54313935
## 370  2.056008476   32.57618944
## 371  2.151914862   31.92020673
## 372 -0.883874672    2.41731919
## 373 -0.050119845   14.78228941
## 374  0.650239066   27.79314103
```

```
## 375 -0.912402648    0.21133187
## 376  0.004489019  -16.26143540
## 377  1.935370053   28.77161469
## 378  0.531276439  -19.88436802
## 379  1.395486905    1.61274444
## 380  1.829241732    4.97771466
## 381  0.729600644  -19.01143372
## 382 -1.733041071  -73.59324288
## 383 -1.916149404  -71.06601015
## 384 -0.785268370  -30.03296006
##
## $ARorder
## [1] 2
##
## $MAorder
## [1] 2
##
## $cnst
## [1] FALSE
##
## $coef
##              [,1]        [,2]
## [1,]  0.760486894  5.87825294
## [2,] -0.012548229  0.53536978
## [3,] -0.060110816 -6.81130054
## [4,]  0.009244218  0.32555532
## [5,] -0.346454644 -6.90663197
## [6,]  0.012013901 -0.06109274
## [7,] -0.036431563  2.67660866
## [8,] -0.001438387 -0.22354678
##
## $secoef
##      [,1] [,2]
## [1,]  NaN  NaN
## [2,]  NaN  NaN
## [3,]  NaN  NaN
## [4,]  NaN  NaN
## [5,]  NaN  NaN
## [6,]  NaN  NaN
## [7,]  NaN  NaN
## [8,]  NaN  NaN
##
## $residuals
##                [,1]        [,2]
##   [1,]  1.937407165  33.06640708
##   [2,] -0.143991888  35.57891151
##   [3,]  2.260555917  39.45015866
##   [4,]  3.227441515  36.41977847
##   [5,] -1.153280412   8.25970462
##   [6,]  1.942138121  23.72630210
##   [7,]  0.513547859   4.47495866
##   [8,]  1.181282008  14.51452845
##   [9,] -0.451850418   2.35129073
##  [10,]  0.801537215  18.16112863
```

```
## [11,]  0.381762781   8.52010602
## [12,] -1.004736969 -17.56802327
## [13,] -0.213381660 -11.97988530
## [14,]  0.707379599  -0.32826783
## [15,] -0.260578225  -7.18826406
## [16,] -0.102757074  23.26228708
## [17,] -0.543760082 -13.60730913
## [18,]  0.935853144  18.28761206
## [19,]  0.275438977  18.58537895
## [20,] -0.840862169 -30.06255508
## [21,]  1.764394851  27.19899997
## [22,]  1.019668114  28.82566737
## [23,] -0.781487273 -25.47302612
## [24,] -1.190649415 -14.11000731
## [25,]  1.030169615  17.81732579
## [26,]  1.799833470  -3.27514114
## [27,] -2.295432576 -12.27947363
## [28,]  2.138608181  36.91890763
## [29,] -0.280293985  -0.48419986
## [30,] -0.574284078  12.64086770
## [31,] -0.355177380  -3.36839643
## [32,] -1.524502879 -32.91229225
## [33,]  1.449465924  17.33852456
## [34,]  0.117680971 -29.48192802
## [35,]  2.603165885  28.74169140
## [36,]  1.400934765  17.56633739
## [37,] -1.806910019 -17.64091379
## [38,] -1.106165767  -6.91219235
## [39,] -1.187793339 -15.45705266
## [40,] -1.639819961 -26.23453816
## [41,] -1.744538829 -30.24125061
## [42,] -1.267663850 -22.97990935
## [43,]  3.083387532  91.51913405
## [44,]  0.022440538  -5.02165075
## [45,] -0.398175519 -20.06704016
## [46,] -0.305337051  -2.62862257
## [47,]  0.488894939   1.19820451
## [48,]  0.602641084  16.03410910
## [49,]  2.241849665  -5.11132748
## [50,]  0.599331215   3.75817949
## [51,] -0.143303546  14.67352843
## [52,] -0.269072386   7.95838643
## [53,]  0.866447143   5.21644823
## [54,]  0.457466637  -8.04324255
## [55,] -1.538044321 -18.79034619
## [56,] -0.861202728   5.01492631
## [57,]  0.758156803   4.67659879
## [58,] -1.122045381 -23.43925028
## [59,] -0.584775612  -8.96546886
## [60,]  1.396813319  23.31556303
## [61,]  1.754978502  24.25892514
## [62,]  0.196921694 -25.05735044
## [63,] -0.781392658  10.16194703
## [64,]  0.383222113  31.19992999
```

```
##  [65,] -0.258616439  16.99099457
##  [66,] -0.511564770  -7.66050172
##  [67,] -0.082166403   6.70200218
##  [68,]  2.240031975   8.59223955
##  [69,]  0.473798420   1.57966024
##  [70,] -0.002731345  21.35256452
##  [71,]  0.339140670   2.88026601
##  [72,] -0.080903055  -9.48468074
##  [73,]  0.722021499 -10.59798916
##  [74,] -1.629298849 -11.75017344
##  [75,] -0.750748480  19.53628600
##  [76,] -0.368407544 -20.01725842
##  [77,] -2.279743121 -53.57265536
##  [78,] -0.633391048   2.65744516
##  [79,]  0.492845690   5.34082184
##  [80,]  0.143942846  14.18991260
##  [81,]  0.347660547  30.48183963
##  [82,]  0.492863531 -49.72107377
##  [83,] -3.050875972 -46.88059970
##  [84,] -1.225258354 -18.54265724
##  [85,]  0.798426543  14.40785658
##  [86,] -0.131564544  -3.26044008
##  [87,] -1.017402954 -11.44092994
##  [88,]  1.863960796  25.91707185
##  [89,]  1.047925748  37.65460937
##  [90,]  0.853744872  52.80531651
##  [91,] -0.497637243  24.68684843
##  [92,]  0.293497663 -20.27323699
##  [93,] -1.244428419 -37.02234608
##  [94,] -0.875981361 -38.73156831
##  [95,] -0.326007797 -16.88972171
##  [96,]  0.566101915  26.36176630
##  [97,]  0.254250768 -15.18350883
##  [98,] -0.026693266   7.54988585
##  [99,]  0.299038622  38.85631428
## [100,] -1.083380814 -31.64693454
## [101,]  1.313021858  -8.06879762
## [102,] -0.305931534  16.92389636
## [103,]  0.571305127   2.70287820
## [104,] -0.475608455   5.74339685
## [105,] -0.750659097 -34.06974664
## [106,] -0.488200093  17.53375712
## [107,] -0.303482320  10.07735910
## [108,] -0.420726223   4.20690361
## [109,]  2.311154993  38.88763905
## [110,]  0.542479296  27.04438183
## [111,] -0.711848031  36.07512944
## [112,] -0.599891437 -21.16868229
## [113,]  0.795089666   1.58634583
## [114,] -1.160929609 -28.51429970
## [115,] -0.575969217 -29.47828788
## [116,]  1.159518330  68.72662749
## [117,]  0.614410279  10.54165513
## [118,] -0.006969208  44.49050917
```

```
## [119,]  0.031482815  -0.51934268
## [120,] -1.142812452  -5.19126938
## [121,] -0.749740825 -40.89808751
## [122,] -0.232978314  -6.20866709
## [123,]  0.508019409 -22.84807399
## [124,]  0.767092899  19.97523284
## [125,] -0.371232102  15.60303447
## [126,]  0.064154791  21.33334318
## [127,]  0.228197875  32.57501650
## [128,] -1.138401376 -34.00682347
## [129,] -0.664532911 -32.73712097
## [130,]  1.104095711  56.91503299
## [131,] -0.338001406   7.27614956
## [132,] -0.741763670   2.68886339
## [133,] -0.143260641  -1.52187582
## [134,] -0.079980476 -20.88425400
## [135,] -0.776703912 -36.05017652
## [136,] -0.964857772 -21.79072559
## [137,]  0.010378872  -2.94276762
## [138,] -1.116367057 -22.94802856
## [139,] -0.615348787  12.23507607
## [140,] -0.418787640   6.53619136
## [141,]  0.067452160   9.91307667
## [142,]  0.334653077  23.69022545
## [143,] -1.120143700 -56.22760072
## [144,] -1.017264472  37.63281310
## [145,]  0.271673694   8.65132677
## [146,] -2.516822185 -90.80230224
## [147,] -1.146490994 -30.38728136
## [148,]  0.153343013 -12.68209867
## [149,] -0.558002192 -22.45368100
## [150,] -0.518495128 -11.45134388
## [151,] -0.471034842 -23.27925248
## [152,] -0.977234961  -2.73839081
## [153,] -2.450607656 -76.95673732
## [154,] -1.293732735 -29.34840789
## [155,] -1.174745814   1.77792005
## [156,] -0.634169433 -20.21124642
## [157,] -0.575762493  -8.86206239
## [158,] -0.727749892  -4.07626700
## [159,] -1.180839663   3.21523734
## [160,] -1.928140321 -56.62266636
## [161,] -1.459013217 -35.47924558
## [162,] -0.627007361 -21.10077859
## [163,] -0.112773879  -7.65885030
## [164,] -0.288933772  -8.41313153
## [165,] -0.959701169 -19.09535235
## [166,] -1.522130303   0.30917470
## [167,] -0.546348112 -41.43321677
## [168,] -1.580323413 -34.82739709
## [169,] -0.702192605   6.66457342
## [170,] -0.309020958 -14.88797895
## [171,] -0.416933398 -13.51380885
## [172,] -1.009342213 -22.94245818
```

```
## [173,] -0.918686406 -16.85299437
## [174,]  0.599724136  28.91119909
## [175,]  0.468688536 -34.90262581
## [176,]  0.493451938  28.09336442
## [177,] -1.212517926 -30.36904747
## [178,]  2.708346388  63.32458527
## [179,] -1.938030376 -45.61210271
## [180,] -0.437990489  -3.77638491
## [181,] -0.344656689  24.47548372
## [182,]  0.597492398  44.79598329
## [183,] -0.414083945  -3.21154222
## [184,] -1.450664933 -63.36583296
## [185,] -0.683708655 -32.35485531
## [186,]  3.031534049  75.45146869
## [187,] -3.158070842 -84.67327458
## [188,]  2.281314526  25.36599965
## [189,] -1.361415176 -43.95215930
## [190,] -0.937258237  -4.91349218
## [191,] -0.604259971   4.33565349
## [192,]  1.536505500  58.43889176
## [193,]  2.290633804  57.48153012
## [194,]  1.406896784   8.92383344
## [195,] -1.164974326  12.67579253
## [196,] -0.650541711 -15.28537348
## [197,]  1.078748853 -21.80175757
## [198,] -1.400455222 -58.57103829
## [199,]  0.174891737   1.49487821
## [200,]  1.765565301  17.30713665
## [201,]  2.181507158  46.60211545
## [202,] -0.243725443  12.84255476
## [203,]  0.574097545  -0.35276640
## [204,]  0.280920461  -4.45777254
## [205,] -2.202740181 -53.08911366
## [206,] -0.868666639 -23.61630768
## [207,] -1.164434441 -29.22941694
## [208,] -1.683886467 -36.19929479
## [209,] -1.861002487 -39.17943269
## [210,]  3.379007162  -6.60318286
## [211,]  0.834937819   0.07505019
## [212,]  0.609030607   5.72862639
## [213,] -0.433648167 -12.66658923
## [214,]  0.468524961 -11.67366456
## [215,] -2.323134780 -30.06163463
## [216,] -2.573313252 -32.33611859
## [217,] -1.720956891 -27.35739032
## [218,] -0.864303647 -16.13138226
## [219,] -0.312552296  -7.32941580
## [220,] -0.423466921  -8.86451301
## [221,] -1.066923305 -20.21443171
## [222,] -1.735123464 -32.99697021
## [223,]  3.879129760  50.42742226
## [224,] -0.023904251 -52.75857312
## [225,]  3.392757919  32.94549358
## [226,] -0.914641558 -17.16347144
```

```
## [227,]  0.895418035   9.61523532
## [228,]  3.328006486  23.37455486
## [229,]  2.538828851  30.13878110
## [230,] -2.927789204 -41.82754771
## [231,]  0.726548729   0.61279660
## [232,]  0.089179513  19.31967642
## [233,]  0.463865416  19.85107062
## [234,]  2.275681031 -38.00749187
## [235,] -1.522822109  -4.79354914
## [236,]  4.026429108  42.23734397
## [237,] -2.787006418 -51.21640171
## [238,]  3.229566596  39.49056204
## [239,]  1.915899880  21.05656672
## [240,] -1.862899934  -6.15058994
## [241,] -0.963898099   2.37556152
## [242,]  2.357151862  59.58735696
## [243,] -1.344912286 -26.29330982
## [244,] -1.092050954   8.77766341
## [245,]  1.697330521  18.53714444
## [246,] -0.006539888  -8.88048665
## [247,] -0.432384749  32.92710816
## [248,] -2.291045370 -71.32708176
## [249,]  0.685062447  44.46551456
## [250,]  0.573661673  25.66755774
## [251,]  4.754700642  76.15369800
## [252,] -1.698776722  41.97504219
## [253,]  3.494983808  -2.50408531
## [254,] -1.247101629  35.91103776
## [255,] -1.726399045   2.65636988
## [256,]  3.039434123  32.23052145
## [257,]  5.834981976  -5.73138221
## [258,] -0.043603460 -47.85751096
## [259,] -2.209672866  -2.19309115
## [260,]  5.318165433  76.65033919
## [261,] -0.697888500  -8.60176487
## [262,] -0.428101978 -41.21920298
## [263,]  1.662949920  72.08290656
## [264,]  0.713107869 -31.66453356
## [265,]  0.878959508  -6.88052650
## [266,]  2.918111967 -18.35928122
## [267,]  0.341601708  11.19880077
## [268,] -1.727546744 -27.35260860
## [269,] -1.575000504 -26.08560696
## [270,]  1.034080471  24.85052660
## [271,] -1.153462759 -13.54582983
## [272,] -1.769537683 -46.09095369
## [273,] -1.418020369 -22.60899979
## [274,]  0.584353331  30.39998838
## [275,] -0.586744097  -7.89733655
## [276,]  3.056443938  39.87517194
## [277,]  4.717991347  91.49332595
## [278,]  1.802847626  32.70484075
## [279,] -0.655472482  -2.35816256
## [280,]  2.264369718  16.18031089
```

```
## [281,] -0.348521416   4.64908646
## [282,] -0.647069344  -9.68092899
## [283,]  1.228053802  -0.54572794
## [284,] -1.007145751 -30.65000223
## [285,]  0.352086214  18.33973021
## [286,] -1.690245987 -23.67250284
## [287,]  4.186859945  63.57206833
## [288,]  1.747543732   8.79825824
## [289,]  0.640841723   5.43256578
## [290,] -1.296722953 -32.06862030
## [291,] -2.120077153 -29.46713588
## [292,] -2.282885593 -49.81716506
## [293,] -2.134603274 -44.47544890
## [294,] -1.569342500 -35.11487683
## [295,] -0.791352121 -19.53460553
## [296,]  2.183919505  56.74072446
## [297,] -0.272521851   9.88930343
## [298,]  0.913569462  16.37044812
## [299,] -0.923368803  -1.86129733
## [300,]  0.452841578  12.49658013
## [301,] -2.801026518 -42.75038848
## [302,]  1.988465435  -0.76995392
## [303,]  1.304815705  22.79751486
## [304,] -0.499215265 -12.26868973
## [305,] -0.907023525 -20.76574779
## [306,]  0.248536843 -22.11118139
## [307,]  0.320693952   8.50759658
## [308,] -0.335704710 -15.37137218
## [309,]  1.091343204  22.68316826
## [310,]  0.727630290  37.01381071
## [311,] -0.603294900 -17.66844806
## [312,]  4.168220160  91.31519058
## [313,] -1.168233052   5.73395450
## [314,] -3.755644080 -82.52665078
## [315,]  1.372776950  14.01236553
## [316,]  1.640179065  43.58098337
## [317,] -1.855177215  44.14853705
## [318,]  0.259867354  -7.85993603
## [319,] -2.147509797 -11.13586501
## [320,] -1.988076898 -37.80676151
## [321,] -2.018157347 -35.33506909
## [322,] -0.671776969  -6.86004435
## [323,] -0.222570217   3.04948594
## [324,] -1.346293345 -54.80714198
## [325,] -0.760267690  -3.72001809
## [326,]  3.816890071 100.75614483
## [327,] -0.903771398 -18.23614511
## [328,]  1.025772185  -4.38361857
## [329,]  0.689944689 100.74097026
## [330,] -1.239541361  42.86134538
## [331,]  0.139553896  54.53000977
## [332,] -1.959363802 -59.46704443
## [333,] -0.770608293  40.16715144
## [334,] -1.235172704 -25.07341483
```

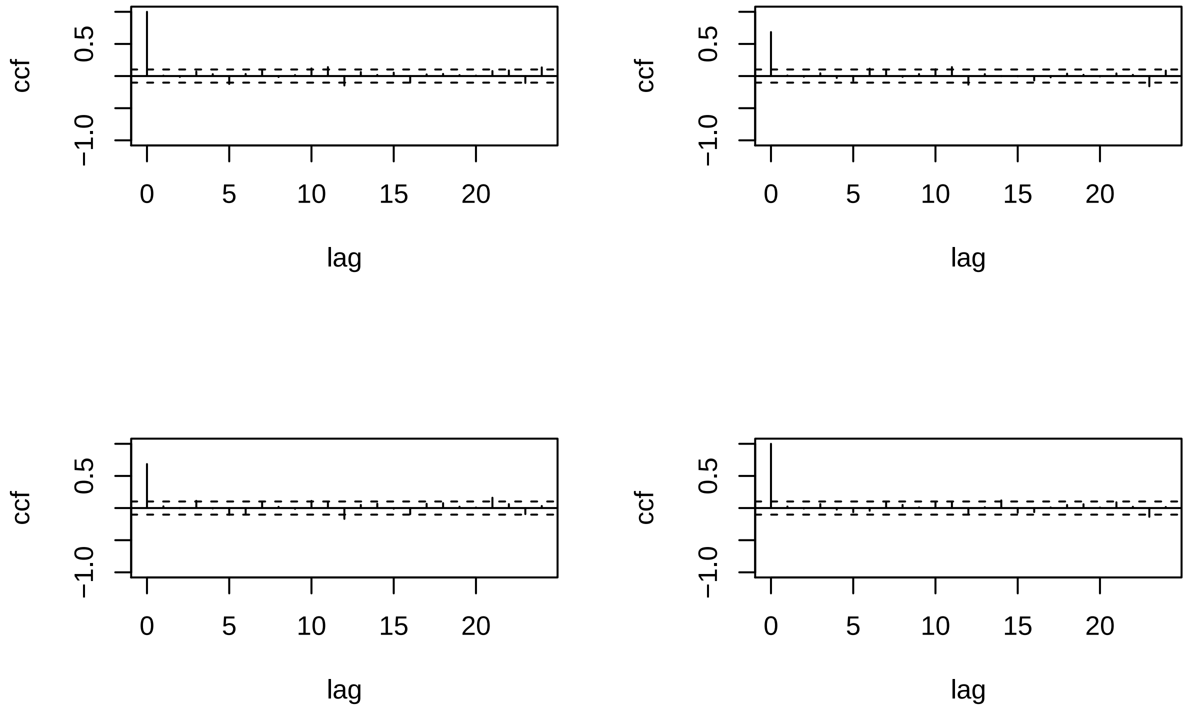
```
## [335,] -0.505614634 -22.31987182
## [336,]  4.003845311  31.17947671
## [337,]  1.598620283  84.92690795
## [338,]  0.804069191  -3.50995422
## [339,] -0.689503092 -15.15397071
## [340,] -0.405988212 -28.27625027
## [341,] -1.950148306 -16.01936455
## [342,]  0.896411006  48.23940648
## [343,] -1.919451675 -23.06744741
## [344,]  0.981817644  17.31646125
## [345,] -0.718659654  20.12585510
## [346,] -1.050117802 -56.01231631
## [347,]  2.763466867  74.61416587
## [348,] -0.184121929  33.32648382
## [349,] -2.671368214 -23.78483532
## [350,] -0.610286959  12.74936871
## [351,]  1.838809513 -28.13763908
## [352,]  0.092445770 -52.02667567
## [353,]  1.007032861  -1.51518664
## [354,] -2.033054250 -48.18964090
## [355,] -1.590798481 -30.52990683
## [356,]  0.716682936  44.22405537
## [357,]  0.306143546   5.97655860
## [358,]  1.370522459  15.09782560
## [359,] -1.442434169  -5.02805508
## [360,] -0.618757450  -0.91510951
## [361,]  0.184181023  19.60763594
## [362,]  0.873670818  16.64265336
## [363,]  0.022236071  -9.23353478
## [364,] -0.359711215   9.94681613
## [365,] -1.020584320   7.67422906
## [366,] -0.394900880   5.03530931
## [367,] -0.317355508   1.11741932
## [368,]  2.518730432  12.08954638
## [369,]  1.468821919   9.33654761
## [370,] -1.791526154 -17.24662368
## [371,]  0.140122074   7.67795279
## [372,]  0.664597816  15.74308596
## [373,] -1.140543832 -16.75112947
## [374,]  0.336128361 -22.79071474
## [375,]  1.995717363  31.40495543
## [376,] -0.135326888 -31.63201778
## [377,]  1.043329019  11.76247440
## [378,]  1.173703936   7.21889303
## [379,] -0.155239714 -15.06481125
## [380,] -2.282163423 -60.38519920
## [381,] -1.394585473 -32.72368972
## [382,]  0.096301527  12.41012618
##
## $Sigma
##          [,1]      [,2]
## [1,]  2.27996  32.42389
## [2,] 32.42389 982.15475
##
```

```
## $aic
## [1] 7.16333
##
## $bic
## [1] 7.32794
##
## $Phi
##           [,1]        [,2]        [,3]        [,4]
## [1,] 0.7604869 -0.01254823 -0.06011082 0.009244218
## [2,] 5.8782529  0.53536978 -6.81130054 0.325555319
##
## $Theta
##           [,1]        [,2]        [,3]        [,4]
## [1,] 0.3464546 -0.01201390  0.03643156 0.001438387
## [2,] 6.9066320  0.06109274 -2.67660866 0.223546781
##
## $Ph0
## [1] 0 0
```

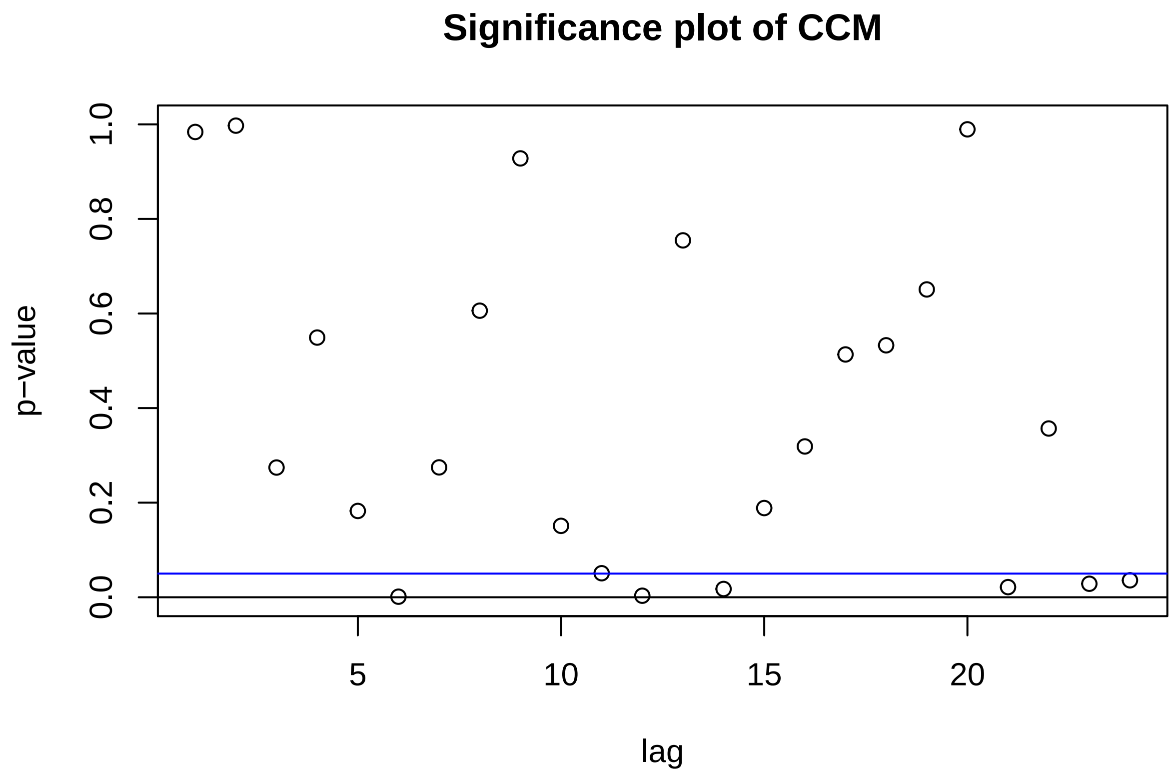
```r
MTSdiag(varma.model)
```

```
## [1] "Covariance matrix:"
##        CO   NO2
## CO   2.29  32.5
## NO2 32.51 984.7
## CCM at lag:  0
##       [,1]  [,2]
## [1,] 1.000 0.685
## [2,] 0.685 1.000
## Simplified matrix:
## CCM at lag:  1
## . .
## . .
## CCM at lag:  2
## . .
## . .
## CCM at lag:  3
## . .
## + .
## CCM at lag:  4
## . .
## . .
## CCM at lag:  5
## - .
## - .
## CCM at lag:  6
## . +
## . .
## CCM at lag:  7
## . +
## . .
## CCM at lag:  8
## . .
## . .
```

```
## CCM at lag:  9
## . .
## . .
## CCM at lag:  10
## + +
## + .
## CCM at lag:  11
## + +
## . .
## CCM at lag:  12
## - -
## - .
## CCM at lag:  13
## . .
## . .
## CCM at lag:  14
## . .
## . +
## CCM at lag:  15
## . .
## . .
## CCM at lag:  16
## - .
## . .
## CCM at lag:  17
## . .
## . .
## CCM at lag:  18
## . .
## . .
## CCM at lag:  19
## . .
## . .
## CCM at lag:  20
## . .
## . .
## CCM at lag:  21
## . .
## + .
## CCM at lag:  22
## . .
## . .
## CCM at lag:  23
## - -
## . -
## CCM at lag:  24
## + .
## . .
```

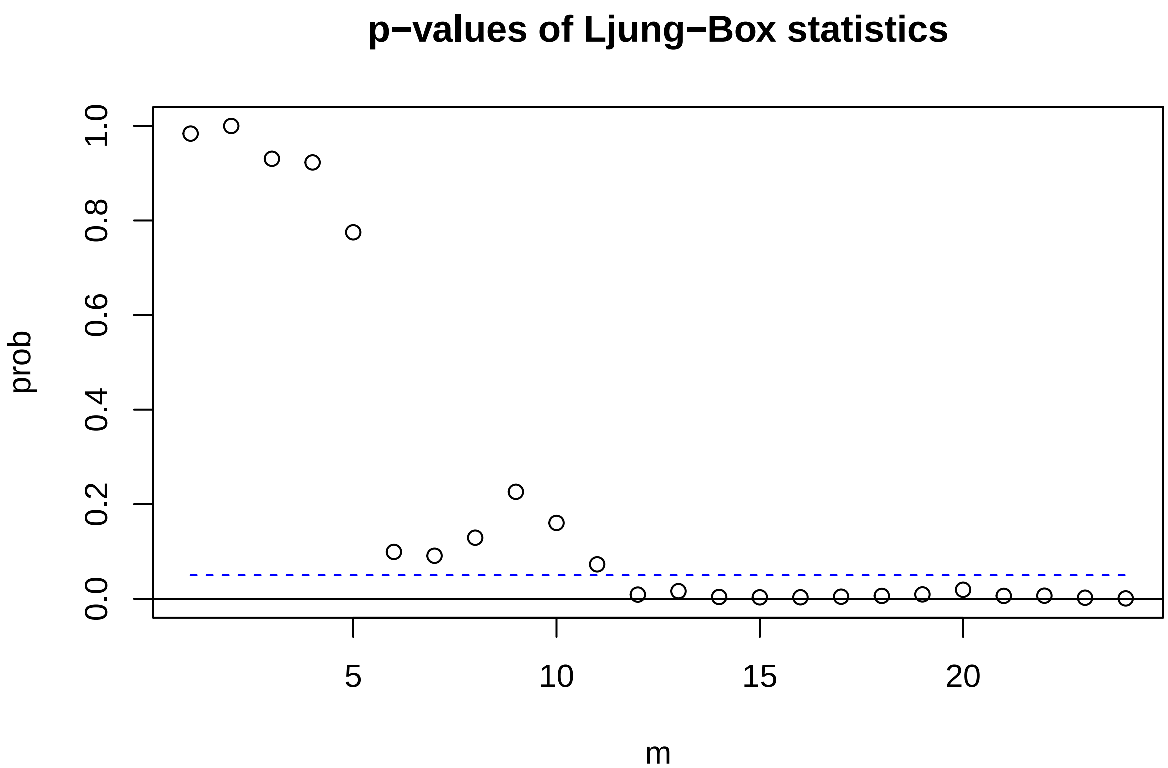



```
## Hit Enter for p-value plot of individual ccm:
```

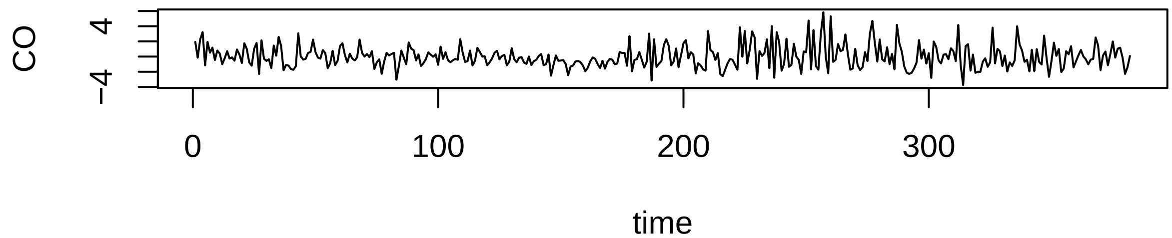


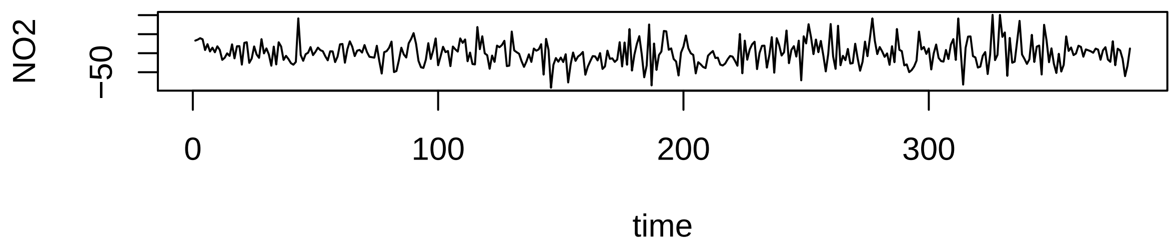

```
## Hit Enter to compute MQ-statistics:
##
## Ljung-Box Statistics:
##           m      Q(m)     df    p-value
##  [1,]   1.000     0.384   4.000     0.98
##  [2,]   2.000     0.538   8.000     1.00
##  [3,]   3.000     5.698  12.000     0.93
##  [4,]   4.000     8.765  16.000     0.92
##  [5,]   5.000    15.026  20.000     0.77
##  [6,]   6.000    33.237  24.000     0.10
##  [7,]   7.000    38.400  28.000     0.09
##  [8,]   8.000    41.136  32.000     0.13
##  [9,]   9.000    42.022  36.000     0.23
## [10,]  10.000    48.794  40.000     0.16
## [11,]  11.000    58.306  44.000     0.07
## [12,]  12.000    74.248  48.000     0.01
## [13,]  13.000    76.154  52.000     0.02
## [14,]  14.000    88.205  56.000     0.00
## [15,]  15.000    94.364  60.000     0.00
## [16,]  16.000    99.101  64.000     0.00
## [17,]  17.000   102.378  68.000     0.00
## [18,]  18.000   105.546  72.000     0.01
## [19,]  19.000   108.030  76.000     0.01
## [20,]  20.000   108.336  80.000     0.02
## [21,]  21.000   119.895  84.000     0.01
## [22,]  22.000   124.289  88.000     0.01
```

```
## [23,]  23.000   135.167  92.000     0.00
## [24,]  24.000   145.510  96.000     0.00
```




```
## Hit Enter to obtain residual plots:
```





The CCF plots shows the correlations are all within the dashed line, meaning that the correlations are not statistically different from 0. The significance plot of CCM shows a few lags that are below the dashed line. While this is not too problematic, we can do better. The plot of p-values for Ljung-Box statistics shows that the model is adequate until around lag 11. The residual plot for CO and NO2 show noise, *yay again!*

After investigating the AIC values and the diagnostic plots of the two comparable VARMA models, we have decided to use VARMA(2,2) due to its lowest AIC value and acceptable diagnostic plots.

Part E: Diagnostics

While we considered the diagnostics to be acceptable, they could be better. For the p-values for Ljung-Box statistic plot, it would be better if there were more adequate lags (i.e. p-values do not become significant until future lags). In the significance plot of CCM, there were a few points below the dashed line. However, these observations do not pose any major concerns for the remainder of our analysis.

Part 3: Simulating from Univariate and Multivariate Time Series Models

```
next.year.time <- c(1:(365))
next.year <- data.frame(time = next.year.time)

COmean <- predict(CO.trend.seasonal, newdata = next.year)
NO2mean <- predict(NO2.trend.seasonal, newdata = next.year)

set.seed(10)
T.simUCO = arima.sim(CO.ar3$model,365)
```

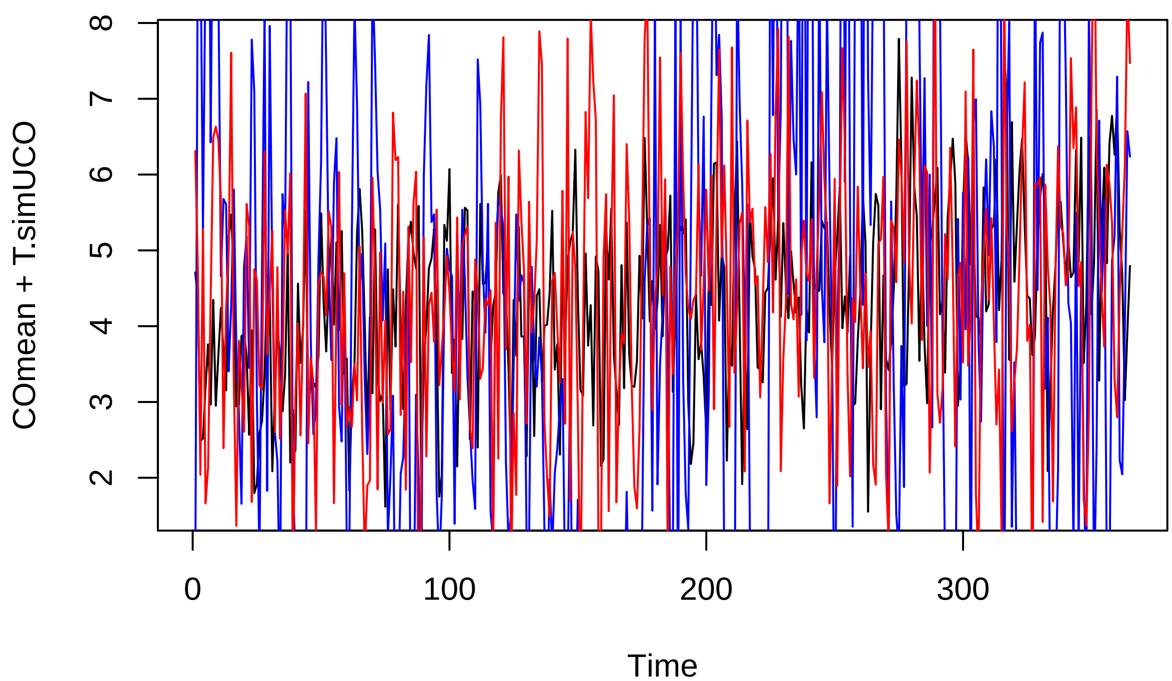
```r
set.seed(5)
T.simUNO2 = arima.sim(n=365, list(ar=c(NO2.ar3$coef[1],NO2.ar3$coef[2],NO2.ar3$coef[3])),sd=sqrt(NO2.ar3
```

```r
T.simM = VARMAsim(365,phi=varma.model$Phi,theta=varma.model$Theta,sigma=varma.model$Sigma)
```

Part A: Ability to reproduce appearance

Multivariate and Univariate: CO

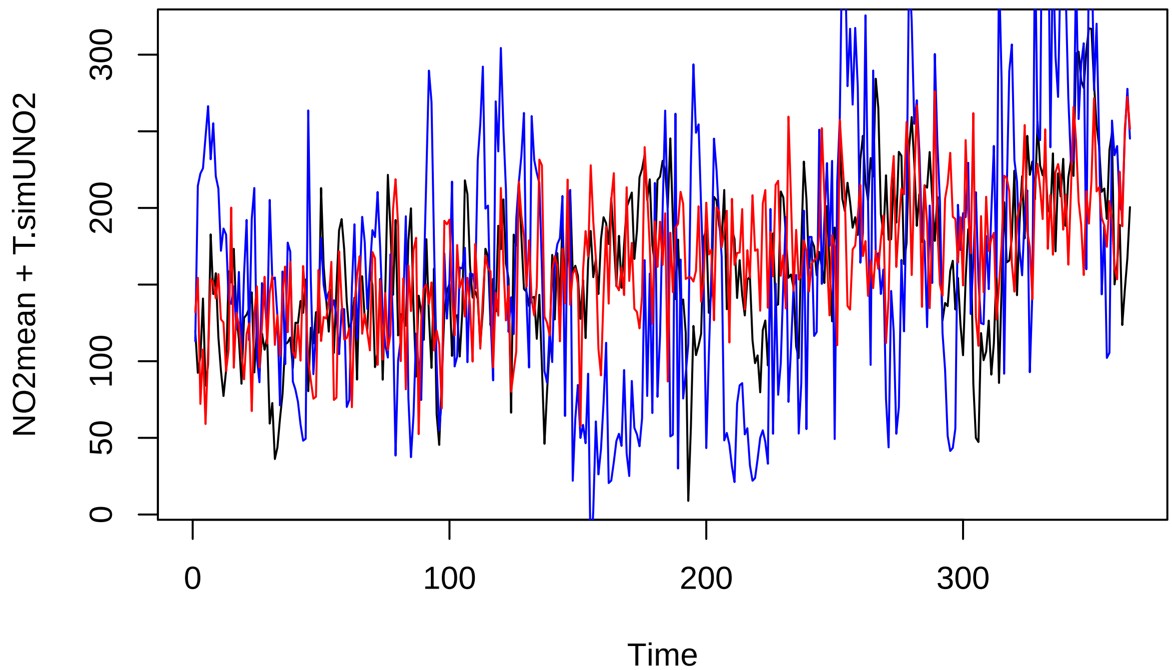
```r
{plot(COmean + T.simUCO)
lines(dailyAQ$CO.GT.[1:365] + allResiduals$CO[1:365],col="blue")
lines(COmean + T.simM$series[,1], col = "red")}
```



Above is a plot of the original CO observations, our univariate simulation of CO, and our multivariate simulation of CO. The plots seem to demonstrate similar behavior, indicating that we were able to reproduce appearance of time series. The one main difference in the graphs is that the magnitudes of the simulated models are slightly greater than the magnitudes of the original observations.

Multivariate and Univariate: NO2

```r
{plot(NO2mean + T.simUNO2)
lines(dailyAQ$NO2.GT.[1:365] + allResiduals$NO2[1:365],col="blue")
lines(NO2mean + T.simM$series[,2], col = "red")}
```



Above is a plot of the original NO2 observations, our univariate simulation of NO2, and our multivariate simulation of NO2. The plots seem to demonstrate similar behavior, however, some areas of the simulated points have slightly different appearance to the original time series (dips in the graph don't completely line up). However, for the most part they do look quite similar, indicating that our simulated models did appropriately reproduce the appearance of time series.

Part B: Ability to reproduce observed trends

Univariate: CO

```
COsim<-COmean+T.simUCO
CO.trend.seasonal.sim <- lm(COsim[next.year.time] ~ next.year.time + sin(2*pi*next.year.time/7) + cos(2*
summary(CO.trend.seasonal)
```

```
##
## Call:
## lm(formula = CO.ts[time] ~ time + sin(2 * pi * time/7) + cos(2 *
##     pi * time/7))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4604 -1.1866 -0.1247  1.0272  6.9821
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            3.7979449  0.1805339  21.037  < 2e-16 ***
```

```
## time                  0.0029483  0.0008127   3.628 0.000325 ***
## sin(2 * pi * time/7) 0.8531164  0.1272445   6.705 7.33e-11 ***
## cos(2 * pi * time/7) 0.3563039  0.1275659   2.793 0.005485 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.765 on 380 degrees of freedom
## Multiple R-squared:  0.1466, Adjusted R-squared:  0.1399
## F-statistic: 21.76 on 3 and 380 DF,  p-value: 5.027e-13
```

```r
summary(CO.trend.seasonal.sim)
```

```
##
## Call:
## lm(formula = COsim[next.year.time] ~ next.year.time + sin(2 *
##     pi * next.year.time/7) + cos(2 * pi * next.year.time/7))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.57100 -0.64785  0.02936  0.73079  2.47035
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   3.5449070  0.1015851  34.896  < 2e-16 ***
## next.year.time                0.0043276  0.0004811   8.996  < 2e-16 ***
## sin(2 * pi * next.year.time/7) 0.6690706  0.0716601   9.337  < 2e-16 ***
## cos(2 * pi * next.year.time/7) 0.2990720  0.0717052   4.171  3.8e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9683 on 361 degrees of freedom
## Multiple R-squared:  0.3389, Adjusted R-squared:  0.3334
## F-statistic: 61.69 on 3 and 361 DF,  p-value: < 2.2e-16
```

The coefficient estimates for time and the seasonality components are very similar to each other for each respective model. This means means that using our simulated data, we were able to reproduce the observed trends of each time series.

Univariate: NO2

```r
NO2sim<-NO2mean+T.simUNO2
NO2.trend.seasonal.sim <- lm(NO2sim[next.year.time] ~ next.year.time + sin(2*pi*next.year.time/7) + cos
summary(NO2.trend.seasonal)
```

```
##
## Call:
## lm(formula = NO2.ts[time] ~ time + sin(2 * pi * time/7) + cos(2 *
##     pi * time/7))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -101.641  -28.675    1.226   26.385  135.816
##
```

```
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          114.9221     4.2424  27.089  < 2e-16 ***
## time                   0.2578     0.0191  13.498  < 2e-16 ***
## sin(2 * pi * time/7)  15.7600     2.9902   5.271 2.28e-07 ***
## cos(2 * pi * time/7)   5.5152     2.9977   1.840   0.0666 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 41.48 on 380 degrees of freedom
## Multiple R-squared:  0.3576, Adjusted R-squared:  0.3525
## F-statistic:  70.5 on 3 and 380 DF,  p-value: < 2.2e-16
```

```
summary(NO2.trend.seasonal.sim)
```

```
##
## Call:
## lm(formula = NO2sim[next.year.time] ~ next.year.time + sin(2 *
##     pi * next.year.time/7) + cos(2 * pi * next.year.time/7))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -139.898  -24.940    4.013   27.070  117.805
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    114.78962    4.42658  25.932  < 2e-16 ***
## next.year.time                   0.25562    0.02096  12.194  < 2e-16 ***
## sin(2 * pi * next.year.time/7)  14.07454    3.12259   4.507 8.88e-06 ***
## cos(2 * pi * next.year.time/7)  10.23274    3.12456   3.275  0.00116 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42.19 on 361 degrees of freedom
## Multiple R-squared:  0.3324, Adjusted R-squared:  0.3269
## F-statistic: 59.92 on 3 and 361 DF,  p-value: < 2.2e-16
```

The coefficient estimates for time and the seasonality components are very similar to each other for each respective model. This means means that using our simulated data, we were able to reproduce the observed trends of each time series.

Multivariate: CO

```
MCOsim<-COmean+T.simM$series[,1]
MCO.trend.seasonal.sim <- lm(MCOsim[next.year.time] ~ next.year.time + sin(2*pi*next.year.time/7) + cos
summary(CO.trend.seasonal)
```

```
##
## Call:
## lm(formula = CO.ts[time] ~ time + sin(2 * pi * time/7) + cos(2 *
##     pi * time/7))
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -3.4604 -1.1866 -0.1247  1.0272  6.9821
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          3.7979449  0.1805339  21.037  < 2e-16 ***
## time                 0.0029483  0.0008127   3.628 0.000325 ***
## sin(2 * pi * time/7) 0.8531164  0.1272445   6.705 7.33e-11 ***
## cos(2 * pi * time/7) 0.3563039  0.1275659   2.793 0.005485 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.765 on 380 degrees of freedom
## Multiple R-squared:  0.1466, Adjusted R-squared:  0.1399
## F-statistic: 21.76 on 3 and 380 DF,  p-value: 5.027e-13
```

```
summary(MCO.trend.seasonal.sim)
```

```
##
## Call:
## lm(formula = MCOsim[next.year.time] ~ next.year.time + sin(2 *
##     pi * next.year.time/7) + cos(2 * pi * next.year.time/7))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.4447 -1.0889  0.0123  0.9868  4.1558
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    3.8841174  0.1683361  23.074  < 2e-16 ***
## next.year.time                 0.0030280  0.0007972   3.798 0.000171 ***
## sin(2 * pi * next.year.time/7) 0.9628113  0.1187475   8.108 8.09e-15 ***
## cos(2 * pi * next.year.time/7) 0.4273001  0.1188223   3.596 0.000368 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.605 on 361 degrees of freedom
## Multiple R-squared:  0.2049, Adjusted R-squared:  0.1983
## F-statistic: 31.01 on 3 and 361 DF,  p-value: < 2.2e-16
```

The coefficient estimates for time and the seasonality components are fairly similar to each other for each respective model. The seasonality coefficients are slightly different, but still close to each other. This means that using our simulated data, we were able to reproduce the observed trends of each time series.

Multivariate: NO2

```
MNO2sim<-NO2mean+T.simM$series[,2]
MNO2.trend.seasonal.sim <- lm(MNO2sim[next.year.time] ~ next.year.time + sin(2*pi*next.year.time/7) + c
summary(NO2.trend.seasonal)
```

```
##
## Call:
## lm(formula = NO2.ts[time] ~ time + sin(2 * pi * time/7) + cos(2 *
```

```
##     pi * time/7))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -101.641  -28.675    1.226   26.385  135.816
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          114.9221     4.2424  27.089  < 2e-16 ***
## time                   0.2578     0.0191  13.498  < 2e-16 ***
## sin(2 * pi * time/7)  15.7600     2.9902   5.271 2.28e-07 ***
## cos(2 * pi * time/7)   5.5152     2.9977   1.840   0.0666 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 41.48 on 380 degrees of freedom
## Multiple R-squared:  0.3576, Adjusted R-squared:  0.3525
## F-statistic:  70.5 on 3 and 380 DF,  p-value: < 2.2e-16
```

```r
summary(MNO2.trend.seasonal.sim)
```

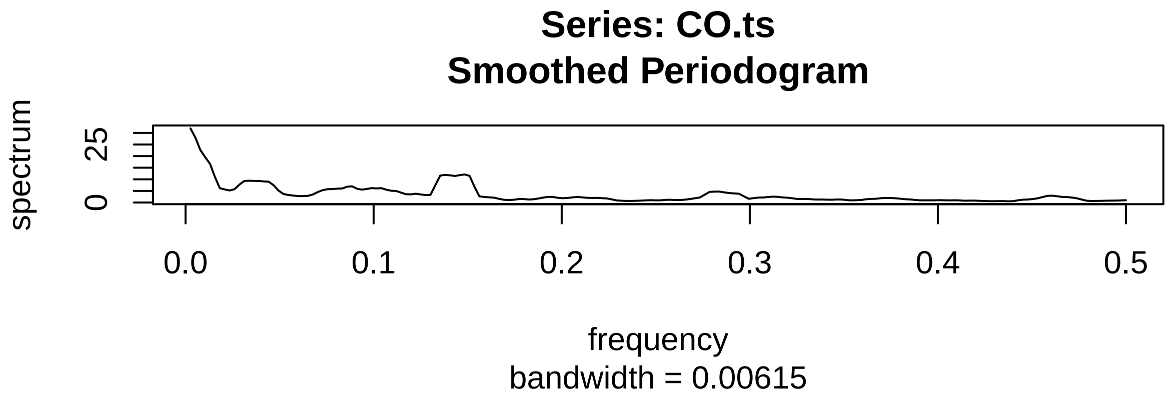
```
##
## Call:
## lm(formula = MNO2sim[next.year.time] ~ next.year.time + sin(2 *
##     pi * next.year.time/7) + cos(2 * pi * next.year.time/7))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -83.063 -22.894   0.474  21.047  88.490
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    117.25399    3.32652  35.248  < 2e-16 ***
## next.year.time                   0.25164    0.01575  15.974  < 2e-16 ***
## sin(2 * pi * next.year.time/7)  18.46201    2.34659   7.868 4.24e-14 ***
## cos(2 * pi * next.year.time/7)   7.23946    2.34807   3.083  0.00221 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.71 on 361 degrees of freedom
## Multiple R-squared:  0.4743, Adjusted R-squared:  0.4699
## F-statistic: 108.5 on 3 and 361 DF,  p-value: < 2.2e-16
```

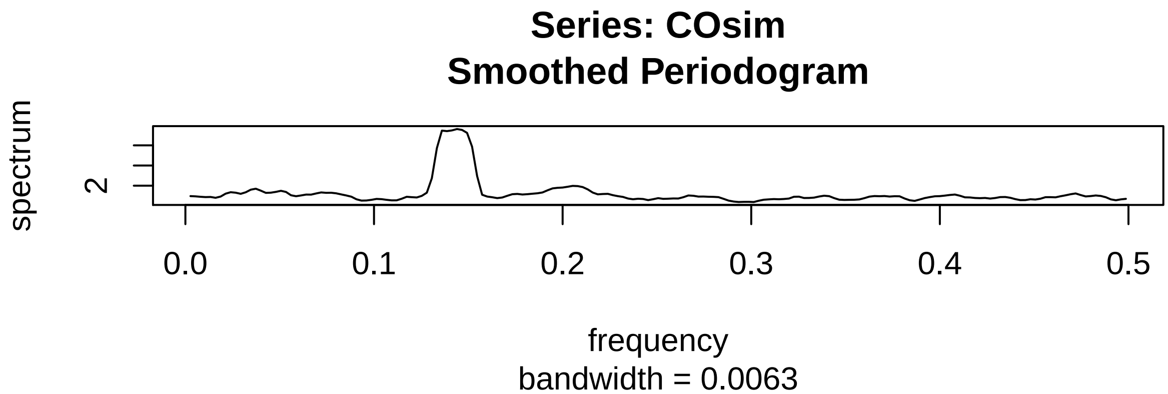
The coefficient estimates for time and the seasonality components are fairly similar to each other for each respective model. The intercept (107.88 vs 114.92) and seasonality coefficients are slightly different, but still close to each other. This means that using our simulated data, we were able to reproduce the observed trends of each time series fairly well.

Part C: Ability to reproduce seasonality

Univariate: CO

```
par(mfrow=c(2,1))
pg.CO <- spec.pgram(CO.ts,spans=9,demean=T,log='no')
pg.COsim <- spec.pgram(COsim,spans=9,demean=T,log='no')
```



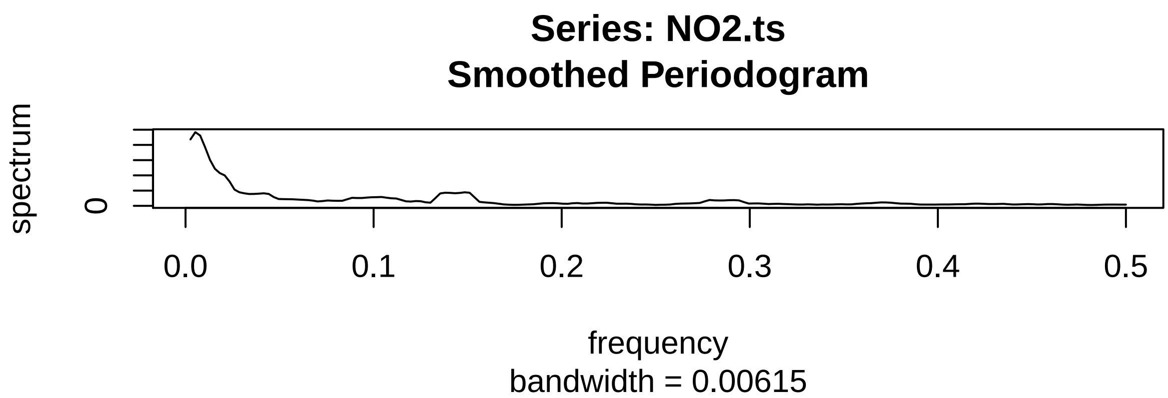


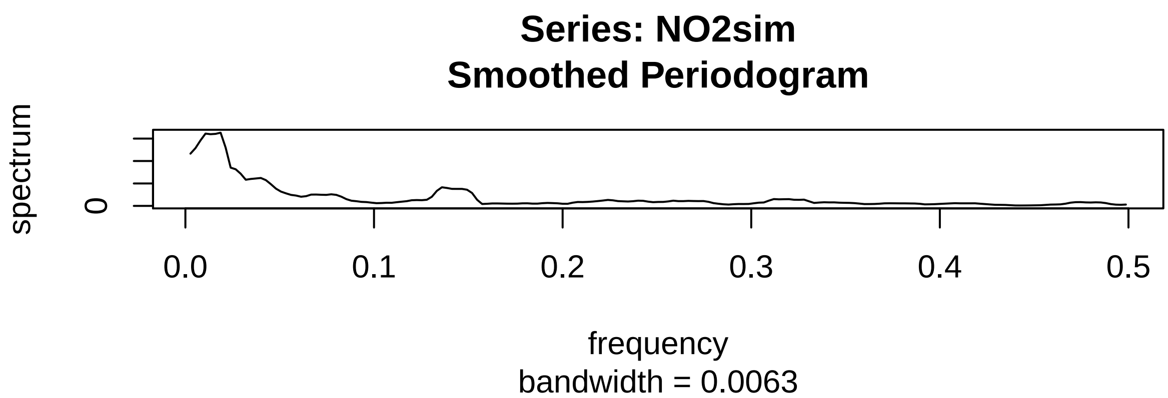

```
par(mfrow=c(1,1))
```

The periodograms show spikes at the same locations and follow the same general trend, indicating that our univariate model was able to reproduce seasonality of the original time series.

Univariate: NO2

```
par(mfrow=c(2,1))
pg.NO2 <- spec.pgram(NO2.ts,spans=9,demean=T,log='no')
pg.NO2sim <- spec.pgram(NO2sim,spans=9,demean=T,log='no')
```



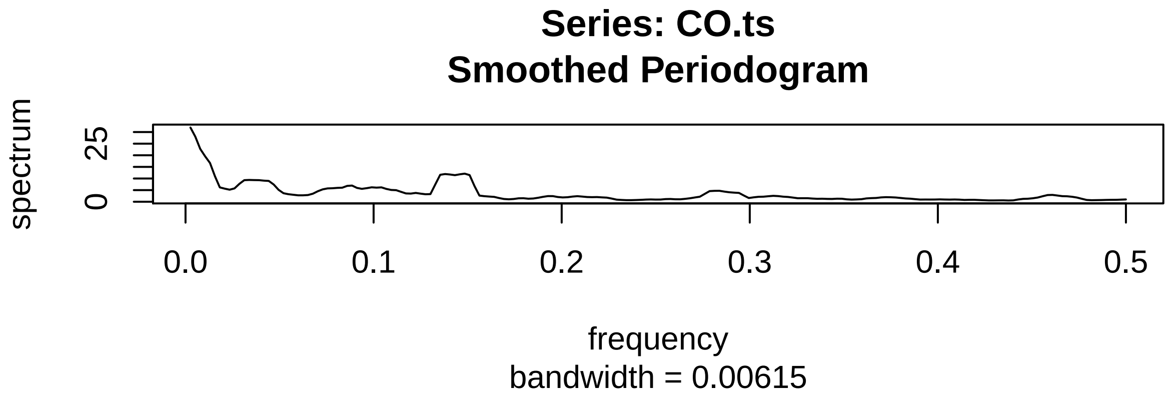


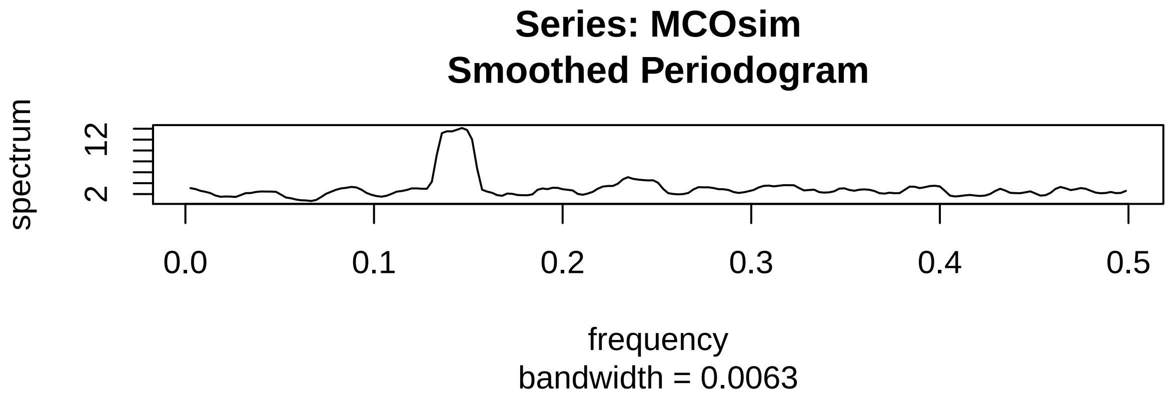

```
par(mfrow=c(1,1))
```

Again, the periodogram of our simulated model has spikes at the same locations as the original time series periodogram, indicating that the univariate model for NO2 was able to reproduce seasonality of the original time series.

Multivariate: CO

```
par(mfrow=c(2,1))
pg.CO <- spec.pgram(CO.ts,spans=9,demean=T,log='no')
pg.MCOsim <- spec.pgram(MCOsim,spans=9,demean=T,log='no')
```



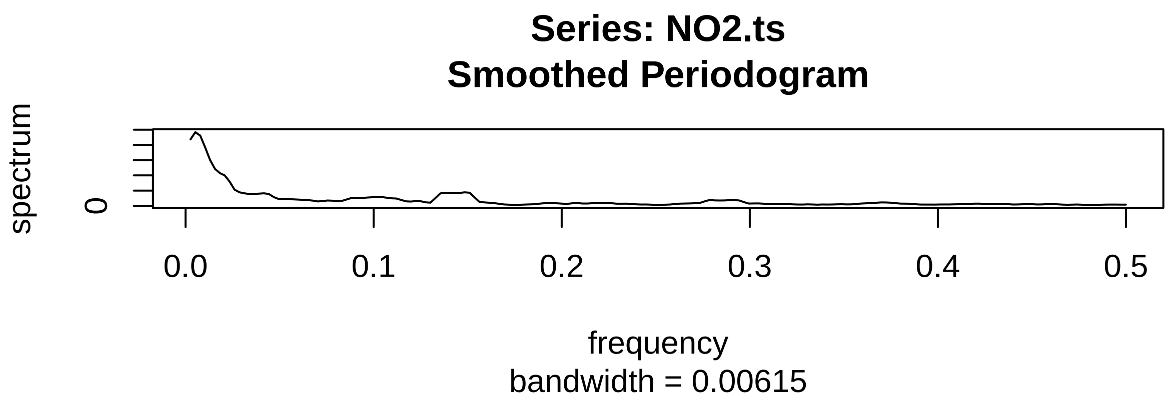


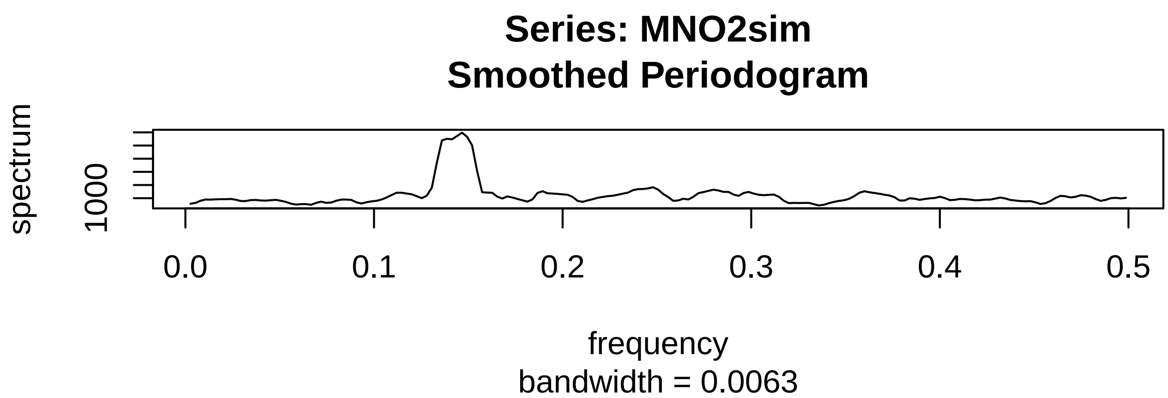

```
par(mfrow=c(1,1))
```

The periodograms look similar and show peaks at the same locations indicating that the simulated multivariate CO model effectively reproduced seasonality.

Multivariate: NO2

```
par(mfrow=c(2,1))
pg.NO2 <- spec.pgram(NO2.ts,spans=9,demean=T,log='no')
pg.MNO2sim <- spec.pgram(MNO2sim,spans=9,demean=T,log='no')
```






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

The periodograms look similar and show peaks at the same locations indicating that the simulated multivariate NO2 model effectively reproduced seasonality.

All of the above simulated periodograms for univariate and multivariate models all show a peak frequency at around 0.14, similar to the observed periodograms. This means we were able to accurately reproduce seasonality of each time series.

Part D: Ability to reproduce observed mean and variance

Univariate: CO

```r
mean(dailyAQ$CO.GT.)
```

```
## [1] 4.364574
```

```r
mean(COsim)
```

```
## [1] 4.338801
```

```r
var(dailyAQ$CO.GT.)
```

```
## [1] 3.622965
```

```r
var(COsim)
```

```
## [1] 1.406698
```

The means are very close to each other (4.365 vs. 4.278), which indicates our model reproduces the observed mean well. However, the variances are slightly different (3.623 vs 1.4479). This coincides with the fact that our simulated values were all slightly lower than each of the observed value.

Univariate: NO2

```r
mean(dailyAQ$NO2.GT.)
```

```
## [1] 164.5335
```

```r
mean(NO2sim)
```

```
## [1] 161.6154
```

```r
var(dailyAQ$NO2.GT.)
```

```
## [1] 2657.722
```

```r
var(NO2sim)
```

```
## [1] 2644.952
```

The means are very close to each other (164.5335 vs. 161.6154), which indicates our model reproduces the observed mean well. The variances are also very close (2657.72 vs 2644.952) indicating that our model also reproduces observed variance well.

Multivariate: CO

```r
mean(dailyAQ$CO.GT.)
```

```
## [1] 4.364574
```

```r
mean(MCOsim)
```

```
## [1] 4.441029
```

```r
var(dailyAQ$CO.GT.)
```

```
## [1] 3.622965
```

```r
var(MCOsim)
```

```
## [1] 3.211575
```

The means are very close to each other (4.365 vs. 4.3988), which indicates our multivariate model reproduces the observed mean for CO well. The variances are also similar (3.623 vs 2.847), indicating that our multivariate model also reproduces the observed variance CO well.

Multivariate: NO2

```r
mean(dailyAQ$NO2.GT.)
```

```
## [1] 164.5335
```

```r
mean(MNO2sim)
```

```
## [1] 163.3567
```

```r
var(dailyAQ$NO2.GT.)
```

```
## [1] 2657.722
```

```r
var(MNO2sim)
```

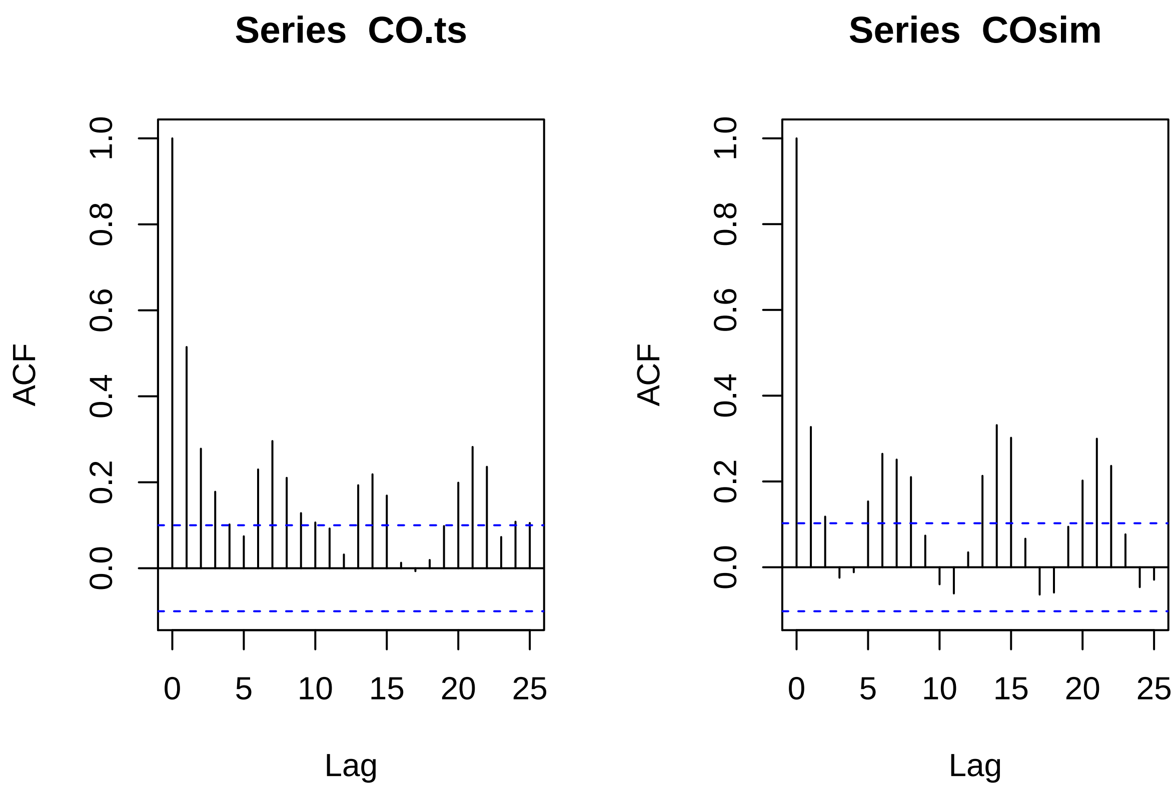
```
## [1] 1896.685
```

The means are very close to each other (164.5335 vs. 160.9884), which indicates our multivariate model reproduces the observed mean for NO2 well. The variances are also fairly similar (2657.722 vs 2056.203), indicating that our multivariate model also reproduces the observed variance for NO2 well.

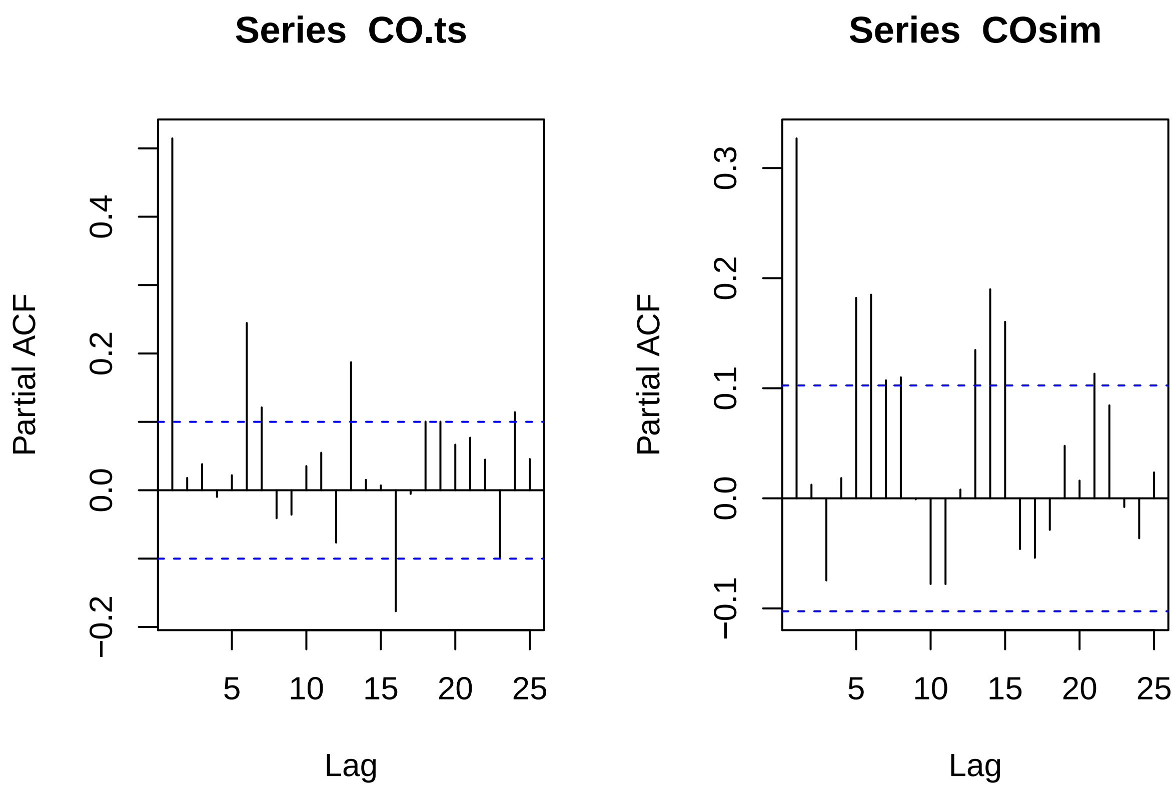
Part E: Ability to reproduce auto-correlation

Univariate: CO

```r
par(mfrow=c(1,2))
acf(CO.ts)
acf(COsim)
```



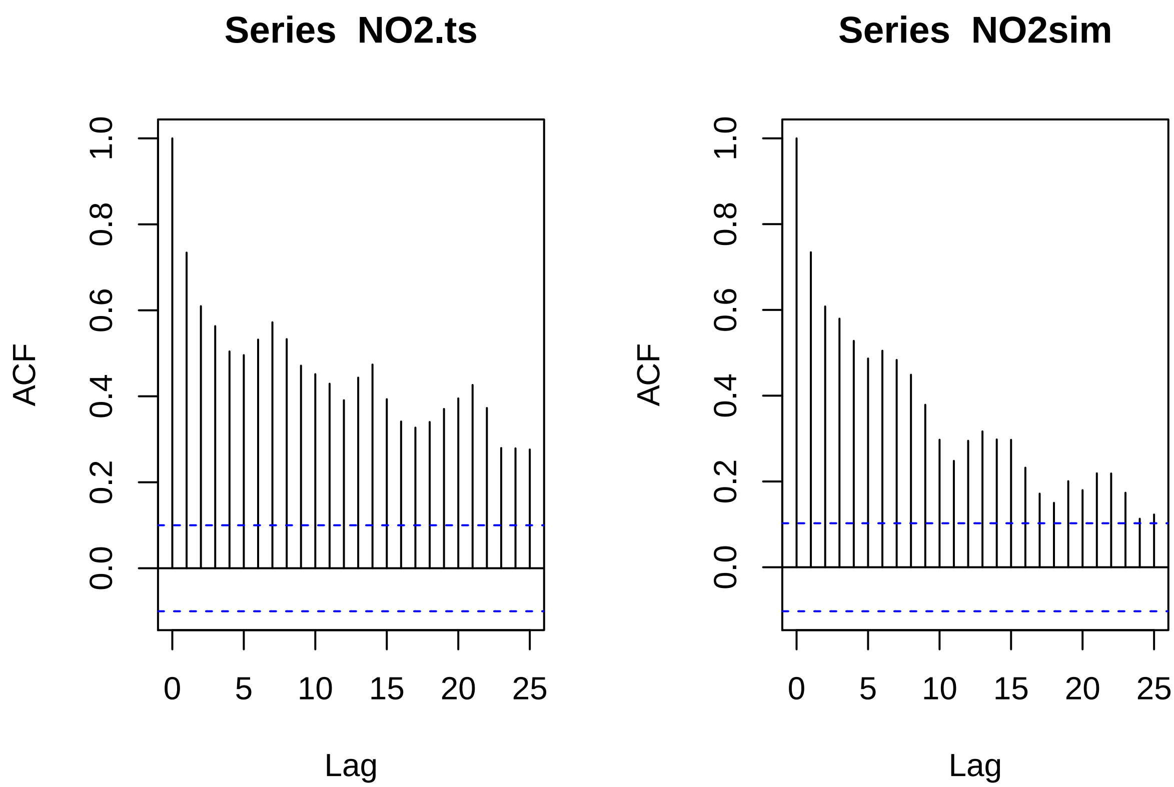

```
par(mfrow=c(1,1))
par(mfrow=c(1,2))
pacf(CO.ts)
pacf(COsim)
```



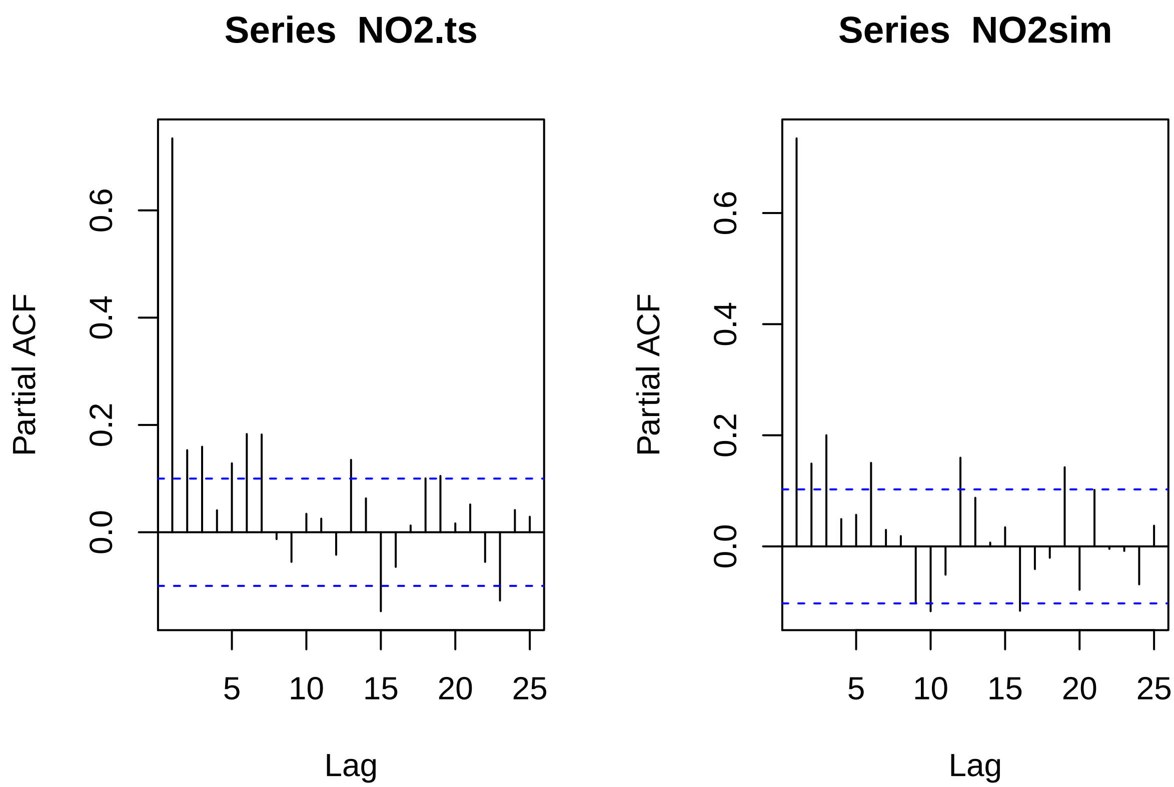

```
par(mfrow=c(1,1))
```

Univariate: NO2

```
par(mfrow=c(1,2))
acf(NO2.ts)
acf(NO2sim)
```



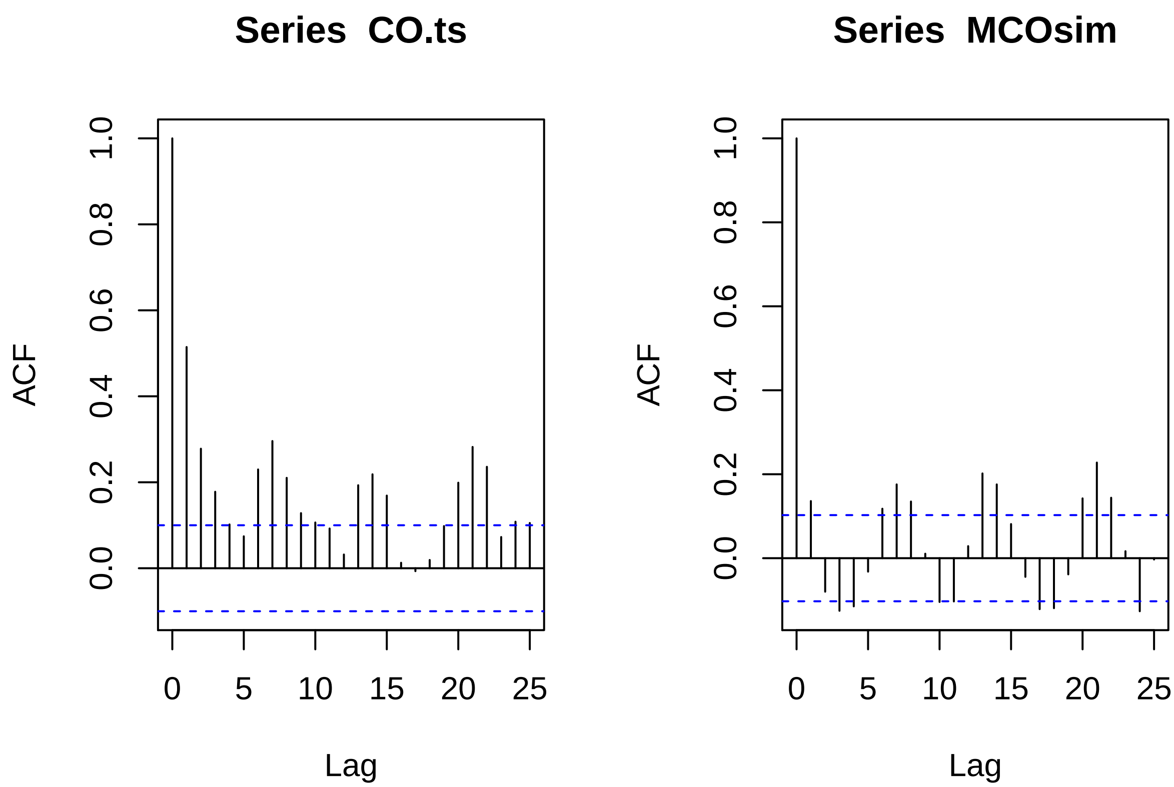

```
par(mfrow=c(1,1))
par(mfrow=c(1,2))
pacf(NO2.ts)
pacf(NO2sim)
```



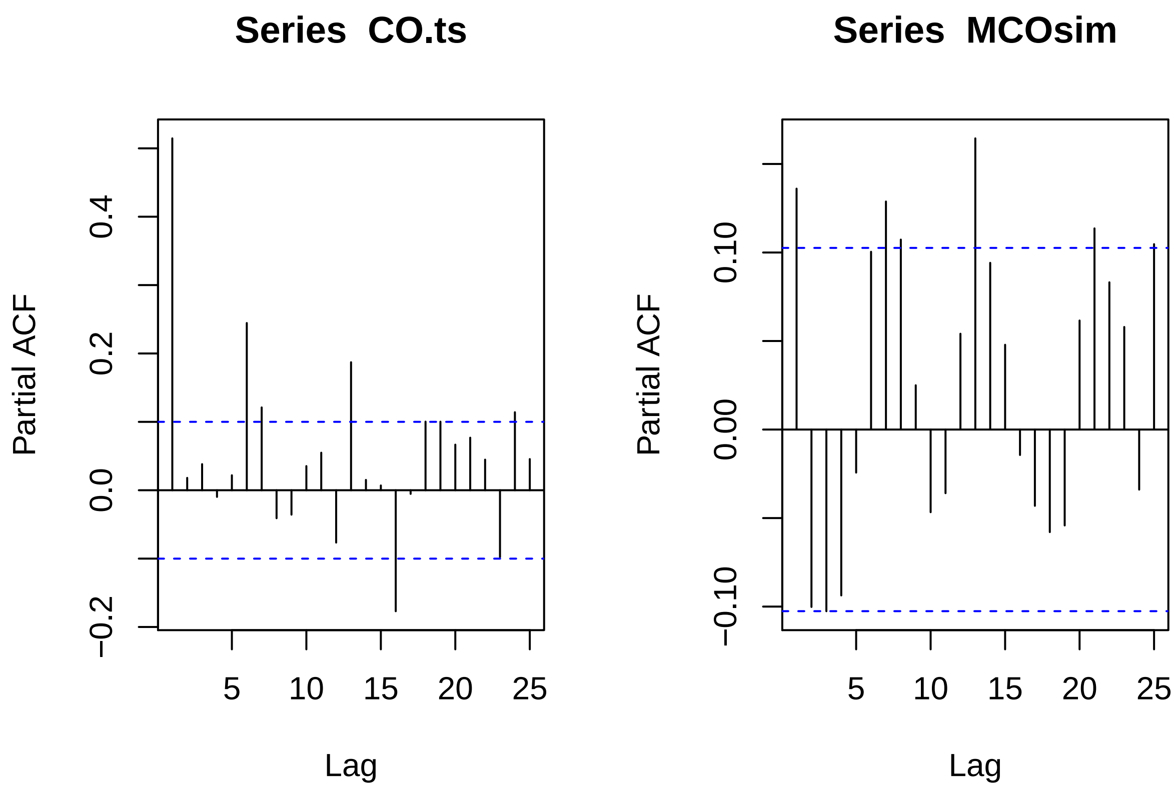

```
par(mfrow=c(1,1))
```

Multivariate: CO

```
par(mfrow=c(1,2))
acf(CO.ts)
acf(MCOsim)
```



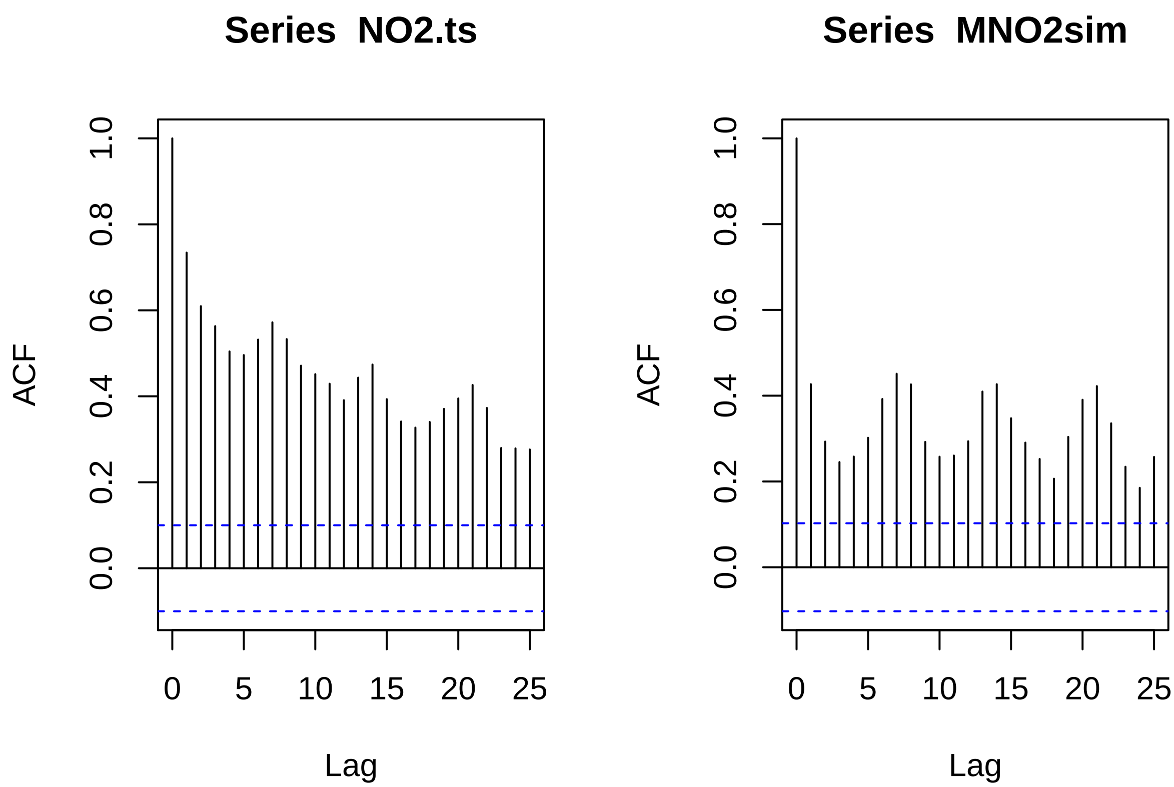

```r
par(mfrow=c(1,1))

par(mfrow=c(1,2))
pacf(CO.ts)
pacf(MCOsim)
```



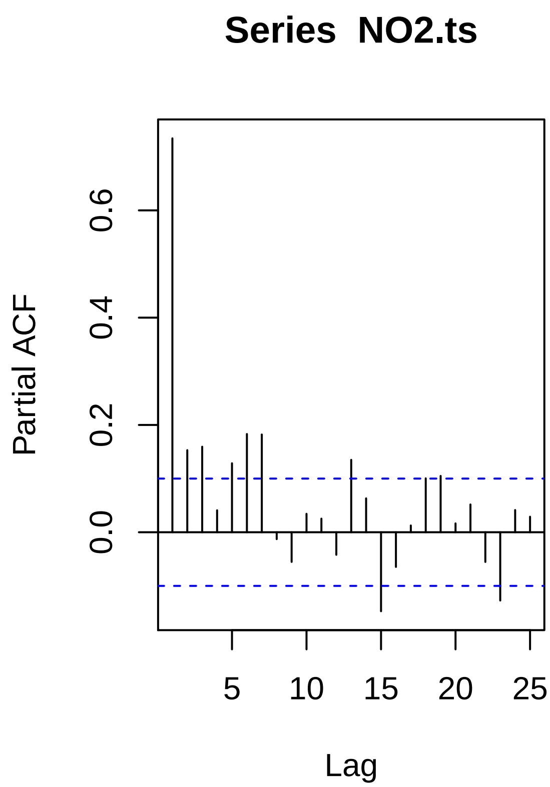

```r
par(mfrow=c(1,1))
```

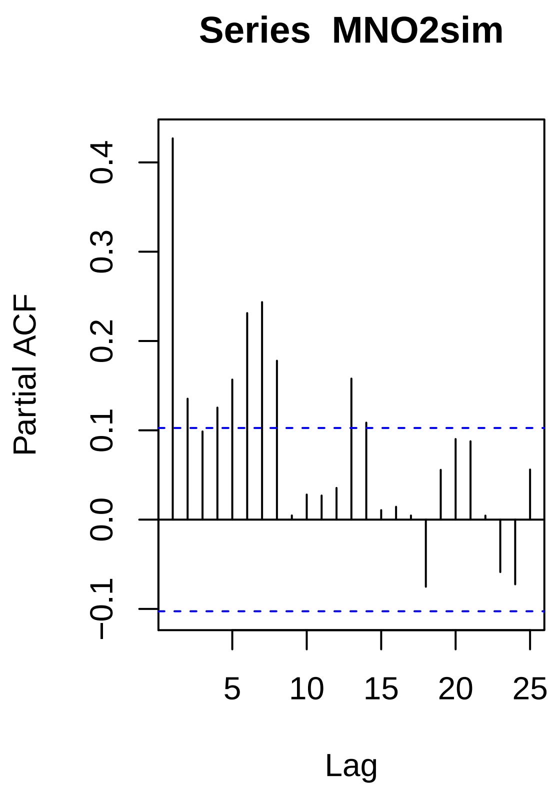
Multivariate: NO2

```r
par(mfrow=c(1,2))
acf(NO2.ts)
acf(MNO2sim)
```




```
par(mfrow=c(1,1))
par(mfrow=c(1,2))
pacf(NO2.ts)
pacf(MNO2sim)
```






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

Despite small differences, the ACF and PACF of each model follows similar trends to those of the original observations. Thus, the models appropriately reproduced the auto-correlation of each time series.

Part F: Ability to reproduce observed cross-correlation

```
cor(dailyAQ$CO.GT.,dailyAQ$NO2.GT.) #observed
```

```
## [1] 0.6076964
```

```
cor(COsim,NO2sim) #simulated from univariate
```

```
## [1] 0.3822696
```

```
cor(MCOsim,MNO2sim) #simulated from multivariate
```

```
## [1] 0.718461
```

The correlation between CO and NO2 in the original data is 0.608. However, the correlation between CO and NO2 in our simulated univariate model is 0.382. So, we would have liked for our univariate models to reproduce observed cross-correlation more accurately. On the other hand, the multivariate simulated model produced a correlation value of 0.718, indicating that the CO and NO2 variables were more correlated than in the original data. However, this difference is much less than the univariate simulation, so we can conclude that the multivariate models were able to more accurately reproduce observed cross-correlations across time series.

Bonus

```
CO.forecast <- forecast(CO.ar3, h=7)
NO2.forecast <- forecast(NO2.ar3, h=7)
MCO.forecast <- forecast(varma.model$data[,1],h=7)
MNO2.forecast <- forecast(varma.model$data[,2], h=7)
```

Prediction performance

Create test set from temp data set with last 6 months

The test period in months

```
next.7day.time <- c((length(dailyAQ)-7):(length(dailyAQ)))
```

The test data frame

```
next.7day.CO <- data.frame(time.temp = 378:384, CO = CO.ts[378:384])
next.7day.NO2 <- data.frame(time.temp = 378:384, NO2 = NO2.ts[378:384])
```

The actual time series for the test period

```
next.7day.CO.ts <- ts(next.7day.CO$CO)
next.7day.NO2.ts <- ts(next.7day.NO2$NO2)
```

Prediction for the next 6 months:

```
E_Y.pred.CO <- predict(CO.trend.seasonal, newdata=next.7day.CO)
```

```
## Warning: 'newdata' had 7 rows but variables found have 384 rows
```

```
e_t.pred.CO <- forecast(CO.ar3, h=7)
next.7day.prediction.CO <- E_Y.pred.CO[378:384] + e_t.pred.CO$mean

e_t.pred.MCO <- forecast(varma.model$data[,1], h=7)
next.7day.prediction.MCO <- E_Y.pred.CO[378:384] + e_t.pred.MCO$mean

E_Y.pred.NO2 <- predict(NO2.trend.seasonal, newdata=next.7day.NO2)
```

```
## Warning: 'newdata' had 7 rows but variables found have 384 rows
```

```
e_t.pred.NO2 <- forecast(NO2.ar3, h=7)
next.7day.prediction.NO2 <- E_Y.pred.NO2[378:384] + e_t.pred.NO2$mean

e_t.pred.MNO2 <- forecast(varma.model$data[,1], h=7)
next.7day.prediction.MNO2 <- E_Y.pred.NO2[378:384] + e_t.pred.MNO2$mean
```

MSE:

```
mean((next.7day.prediction.CO-next.7day.CO$CO)^2)
```

```
## [1] 2.439548
```

```
mean((next.7day.prediction.NO2-next.7day.NO2$NO2)^2)
```

```
## [1] 1268.032
```

```
mean((next.7day.prediction.MCO-next.7day.CO$CO)^2)
```

```
## [1] 2.285644
```

```
mean((next.7day.prediction.MNO2-next.7day.NO2$NO2)^2)
```

```
## [1] 1700.825
```

Plot actual values and predicted values

```
par(mfrow=c(1,2))
# CO forecasts
{plot(ts(next.7day.CO$CO),type='o',ylim=c(0,10))
lines(ts(next.7day.prediction.CO),col='red',type='o')
lines(1:7, E_Y.pred.CO[378:384] + e_t.pred.CO$lower[,2], col = "red", lty = "dashed")
lines(1:7, E_Y.pred.CO[378:384] + e_t.pred.CO$upper[,2], col = "red", lty = "dashed")
lines(ts(next.7day.prediction.MCO),col='green',type='o')
lines(1:7, E_Y.pred.CO[378:384] + e_t.pred.MCO$lower[,2], col = "green", lty = "dashed")
```

```
lines(1:7, E_Y.pred.CO[378:384] + e_t.pred.MCO$upper[,2], col = "green", lty = "dashed")
legend(5,9, legend = c("Actual", "Predicted (univariate)", "Predicted(multivariate)"), lwd = 3, col = c

#NO2 forecasts
{plot(ts(next.7day.NO2$NO2),type='o',ylim=c(0,250))
lines(ts(next.7day.prediction.NO2),col='red',type='o')
lines(1:7, E_Y.pred.NO2[378:384] + e_t.pred.NO2$lower[,2], col = "red", lty = "dashed")
lines(1:7, E_Y.pred.NO2[378:384] + e_t.pred.NO2$upper[,2], col = "red", lty = "dashed")
lines(ts(next.7day.prediction.MNO2),col='green',type='o')
lines(1:7, E_Y.pred.NO2[378:384] + e_t.pred.MNO2$lower[,2], col = "green", lty = "dashed")
lines(1:7, E_Y.pred.NO2[378:384] + e_t.pred.MNO2$upper[,2], col = "green", lty = "dashed")
legend(1,75, legend = c("Actual", "Predicted(univariate)", "Predicted(multivariate)"), lwd = 2, col = c
par(mfrow=c(1,1))}
```