Forecasting Demand

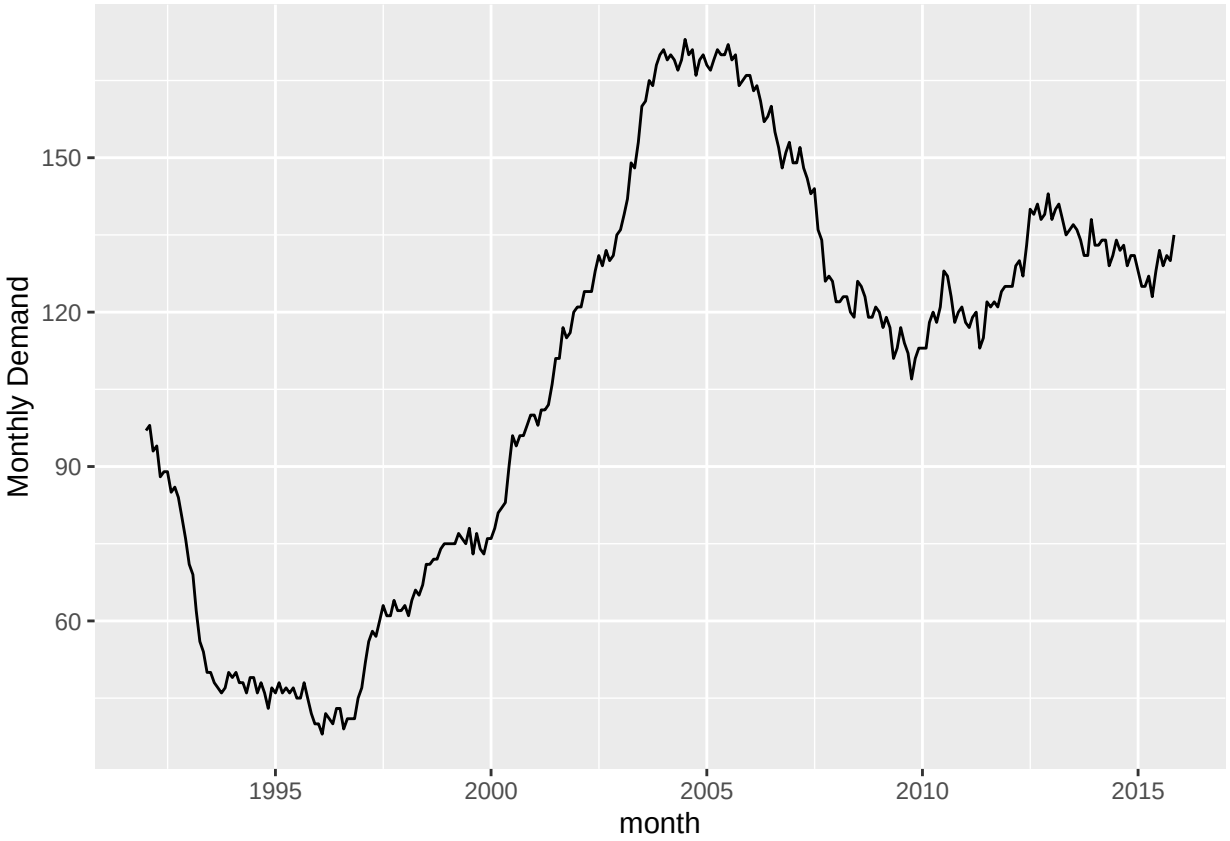
Jacob Kramp

7/30/2021

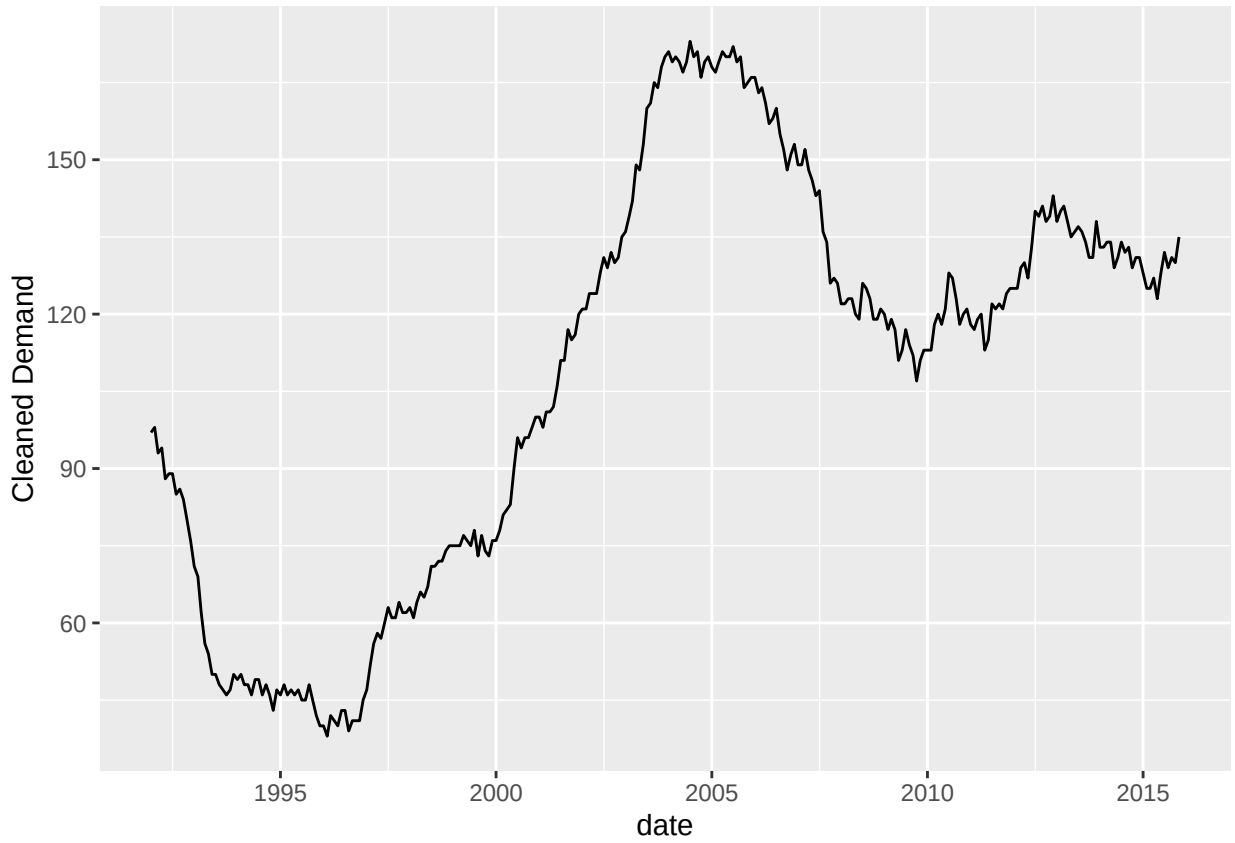
Data Analysis

In order to understand the underlying correlation structure within our time series data, we need to perform some exploratory data analysis. We will begin by plotting the data and assessing if a log transform is necessary due to non-constant variance. We can also check for outliers using the facet_wrap() function.

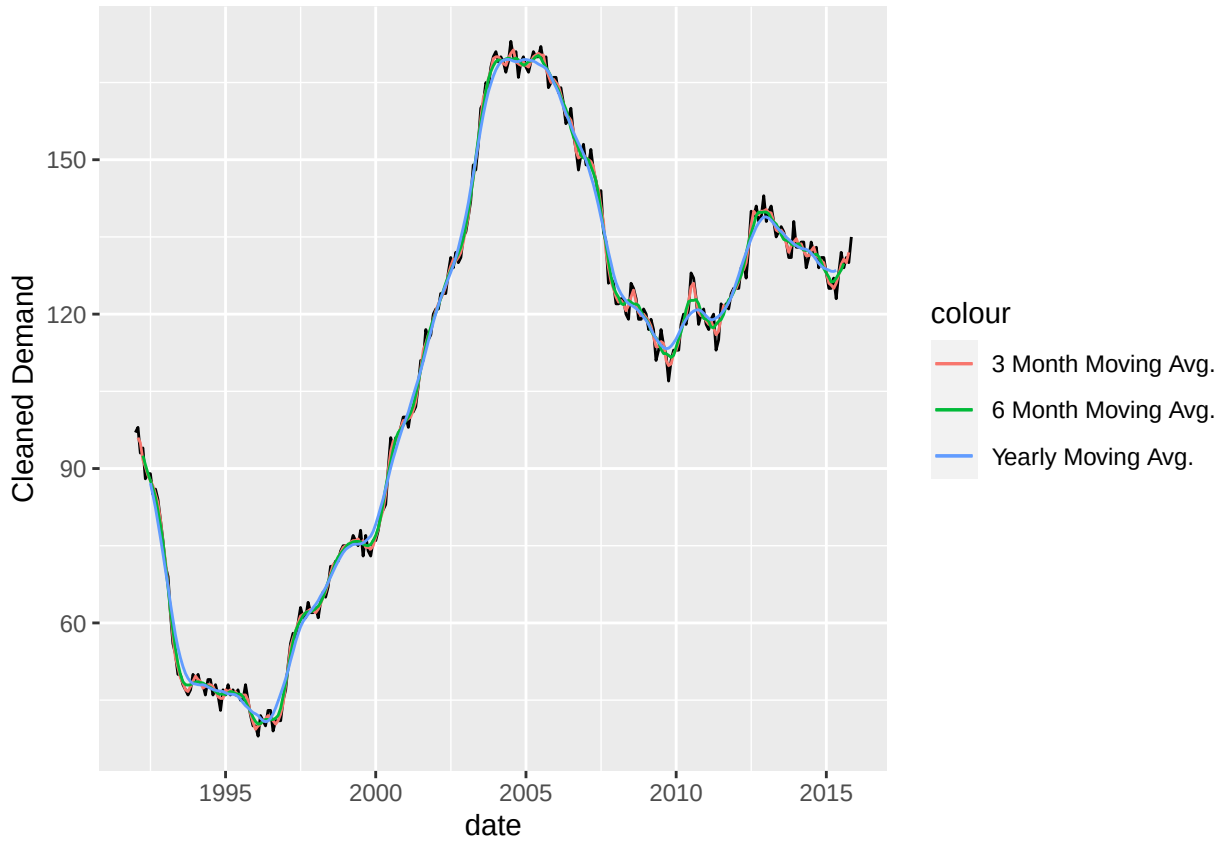
```
#Problem
#A
#EDA
df <- read.csv('~/Math 531T/Demand.txt')
colnames(df) <- 'demand'
dates <- seq(from=as.Date('1992-01-01',format='%Y-%m-%d'),
             to=as.Date('2015-11-01',format='%Y-%m-%d'),by='month')
df$date <- dates
df$month <- months(dates)
df$year <- lubridate::year(dates)
#Plot data
ggplot(df,aes(y=demand,x=date))+geom_line()+scale_x_date('month')+
  ylab('Monthly Demand') +
  xlab('Date')
```




```
#Create a time series object based on demand
count_demand <- ts(df$demand,frequency = 12)
df$clean_demand <- tsclean(count_demand)

#tsClean
ggplot() +
  geom_line(data=df,aes(x=date,y=clean_demand))+ylab('Cleaned Demand')
```

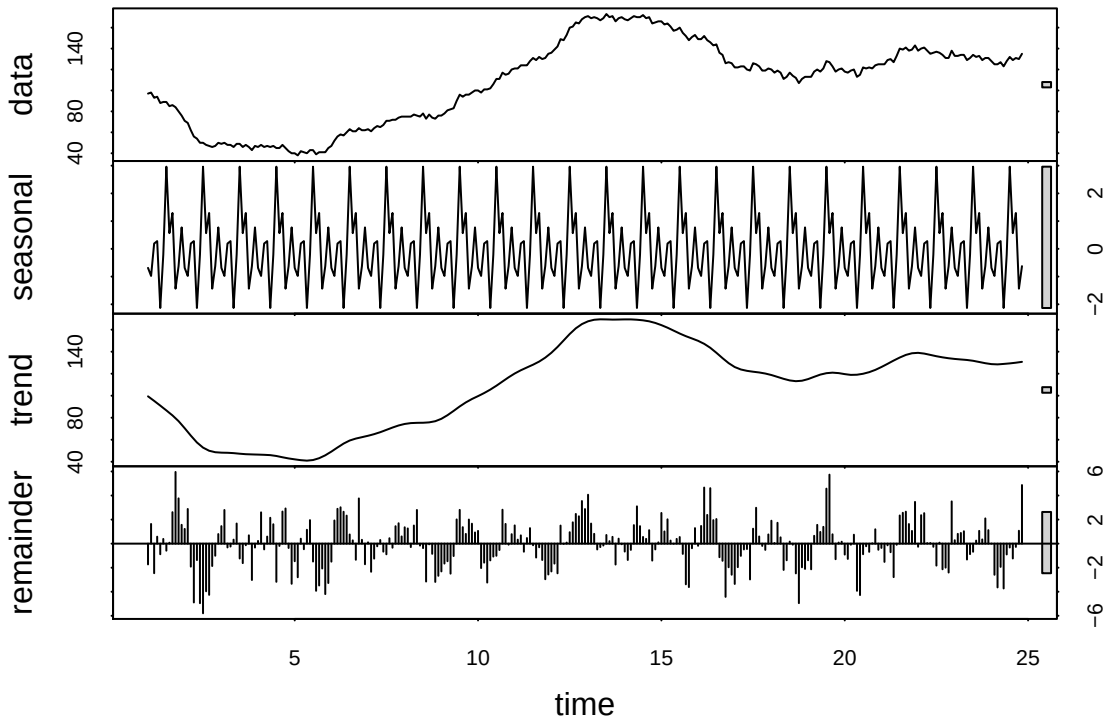



```
#Check out moviung averages
df$ma_3 <- ma(df$clean_demand,order=3)
df$ma_6 <- ma(df$clean_demand,order=6)
df$ma_12 <-ma(df$clean_demand,order=12)

ggplot() +
  geom_line(data=df,aes(x=date,y=clean_demand))+
  geom_line(data=df,aes(x=date,y=ma_3,colour='3 Month Moving Avg.'))+
  geom_line(data=df,aes(x=date,y=ma_6,colour='6 Month Moving Avg.'))+
  geom_line(data=df,aes(x=date,y=ma_12,colour='Yearly Moving Avg.'))+
  ylab('Cleaned Demand')
```



Although our interpretation is subjective, the data doesn't appear to have outliers, but there does seem to be a relatively obvious moving trend throughout the observed time period. There also doesn't seem to be any obvious signs of non constant variance. After smoothing the series using moving averages, the trends throughout become more obvious. There does appear to be some sort of seasonal trend appearing in the data as well. It's difficult to tell, so we can use decomposition functions to get more detail.

```
#Problem B
#decomposition of the data
#ma3 <- ts(na.omit(df$ma_12),frequency = 12)
decomp <- stl(df$clean_demand,s.window='periodic')
decomp1 <-decompose(df$clean_demand)
deseasonal <- seasadj(decomp) #Use later in Arima
plot(decomp)
```



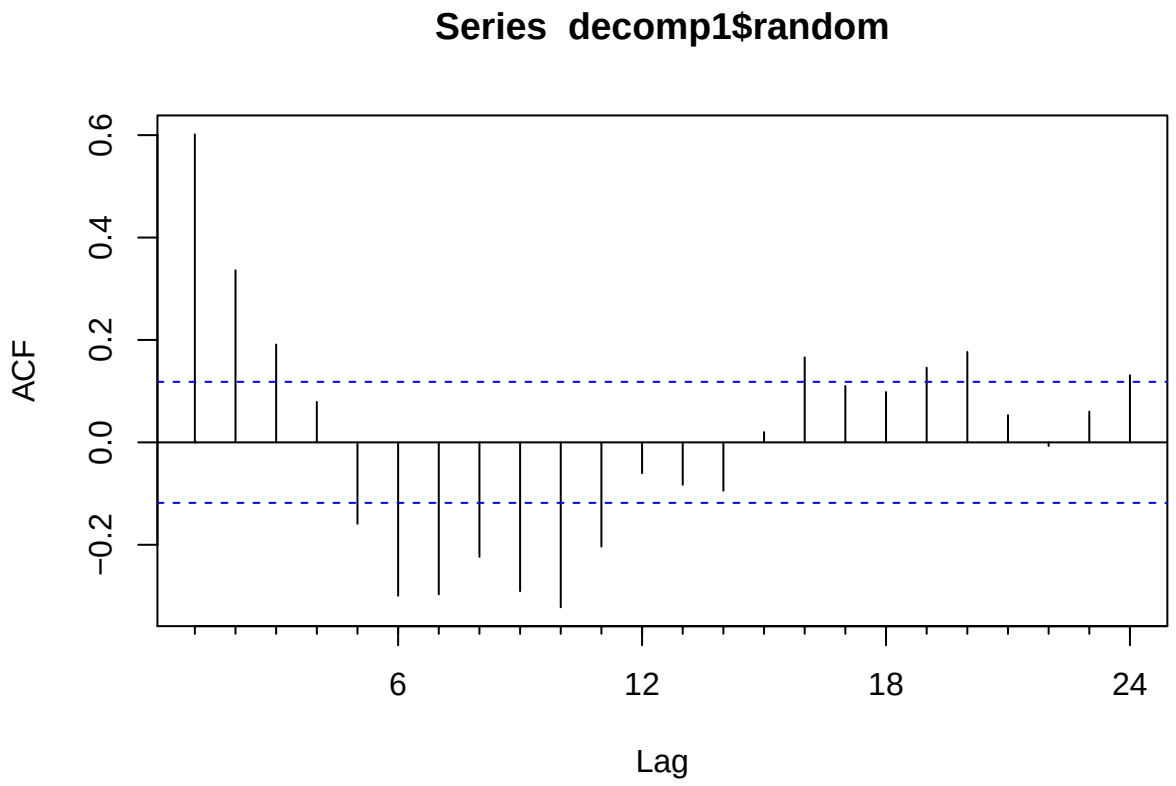

```
adf.test(na.omit(decomp1$random),alternative = 'stationary')
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  na.omit(decomp1$random)
## Dickey-Fuller = -7.6652, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

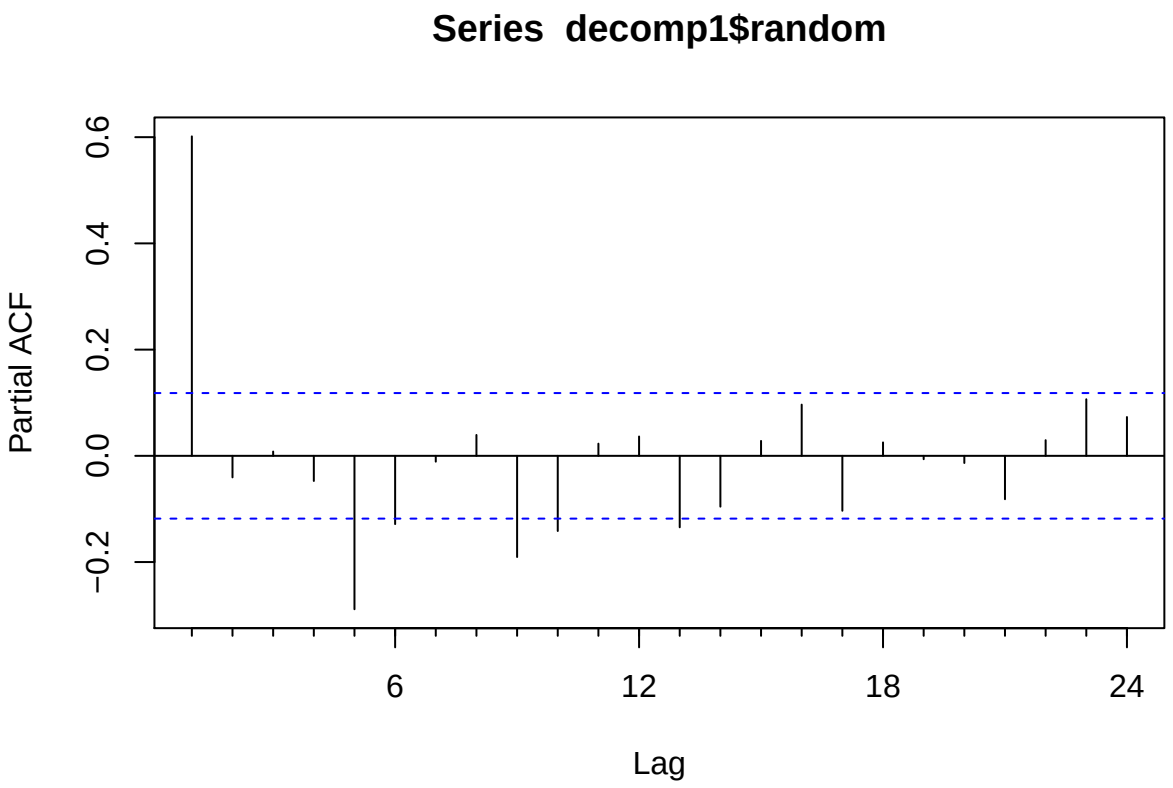
The seasonal trend that we suspected in the "up and down" motion from the previous plot in (a) is more evident in our decomposition seasonal trend plot here. The moving trend is also plotted here very quickly and looks somewhat similar to our moving average plots. Using the Dickey-Fuller test, we can also reject the null hypothesis that our data is non-stationary after decomposing it to remove the trend and seasonal components.

We can now use the ACF and PACF to estimate each parameter we should use in a SARIMA/ARIMA model.

```
#Stationarity testing
#acf(na.omit(decomp1$random))

Acf(decomp1$random)
```



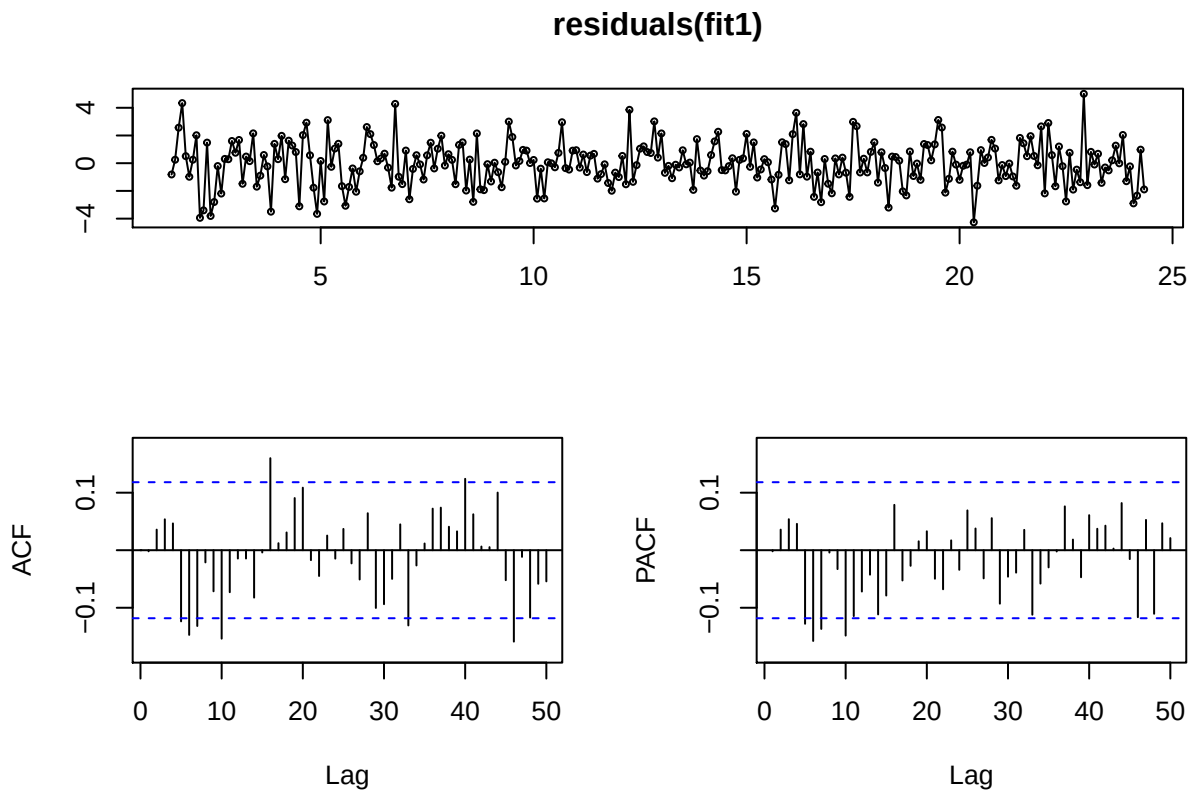

```
Pacf(decomp1$random)
```



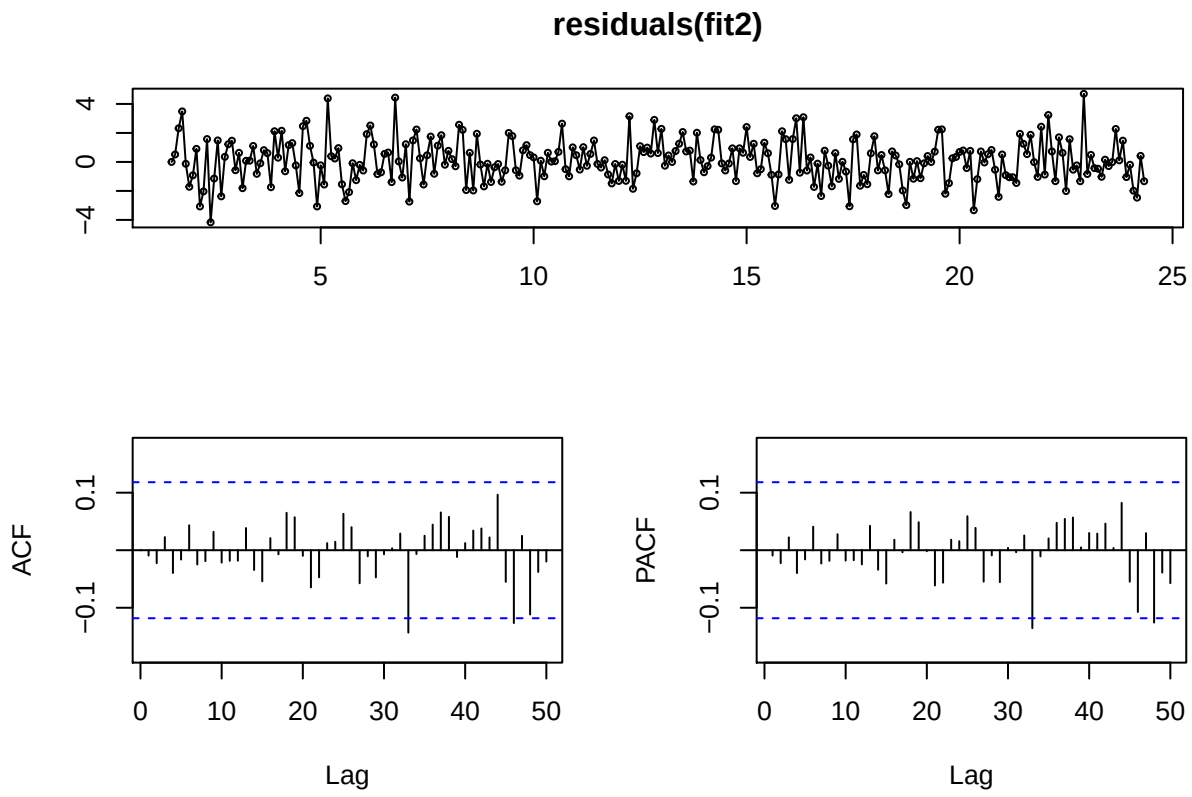

```
#We reject the alternative and accep the null, so our data is not stationary.
```

It seems as though we should use a q=5, d=1 to help with the breakage of the confidence bounds,p = 6, and there seems to be some sort of a bi-annual trend occurring in the data. We can either use a p=3 and P=2 or we I believe we can use p=6 and P=1 to reflect what's happening in the PACF plot. There also appears to be an annual trend, so perhaps S = 12. D=0 and Q = 0 should be okay based on the plots. We can use the auto.arima() function to see if it agrees.

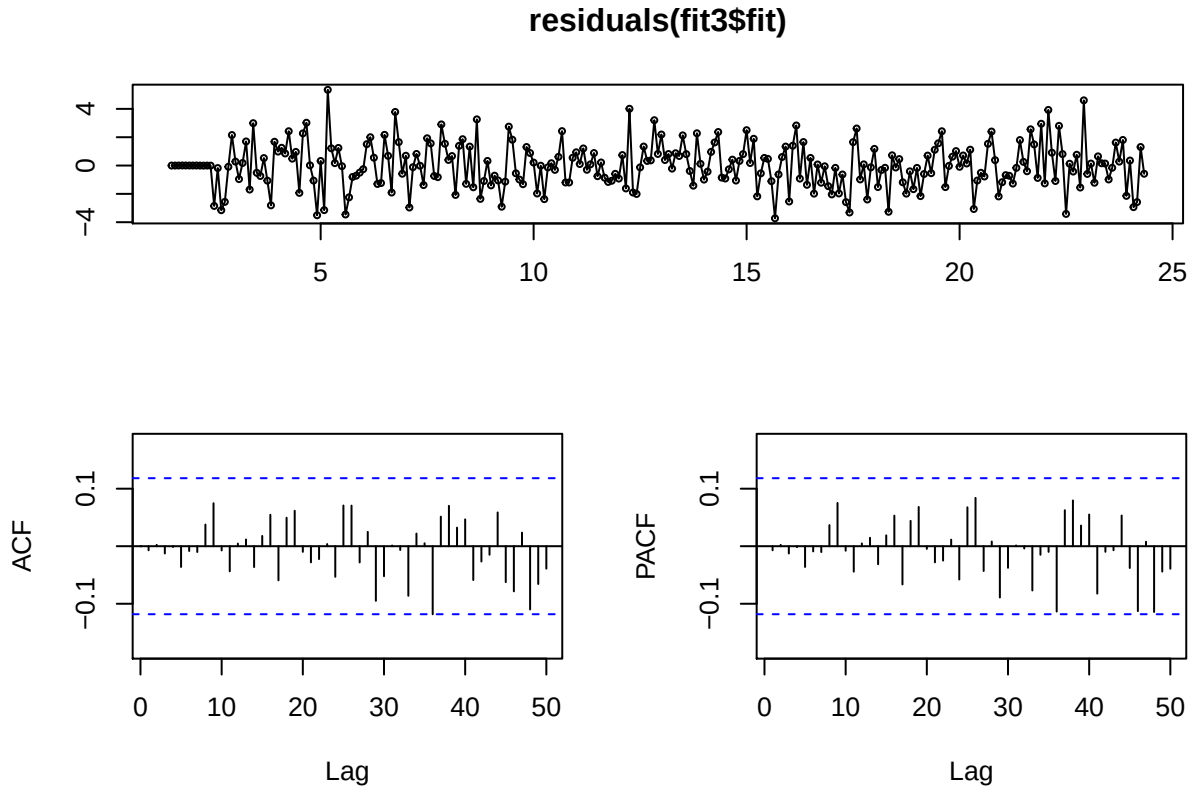
```
fit1 <- auto.arima(na.omit(decomp1$random))
tsdisplay(residuals(fit1),lag.max = 50)
```




```
fit2 <- arima(na.omit(decomp1$random),order = c(6,1,5))
tsdisplay(residuals(fit2),lag.max = 50)
```



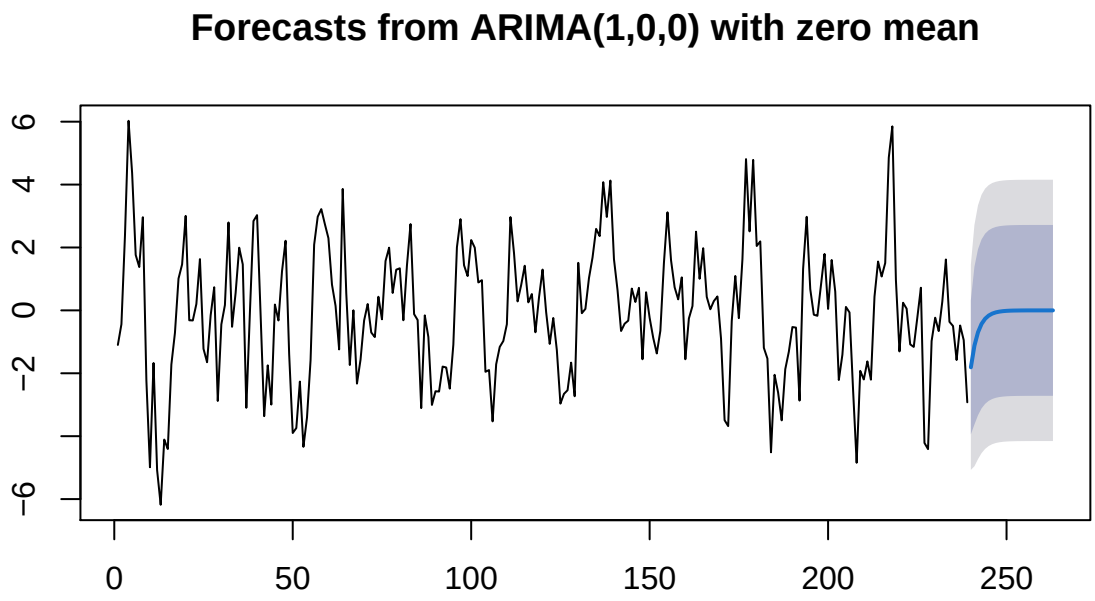

```
fit3 <- sarima(na.omit(decomp1$random),3,0,5,2,1,0,12,details=F)
tsdisplay(residuals(fit3$fit),lag.max = 50)
```

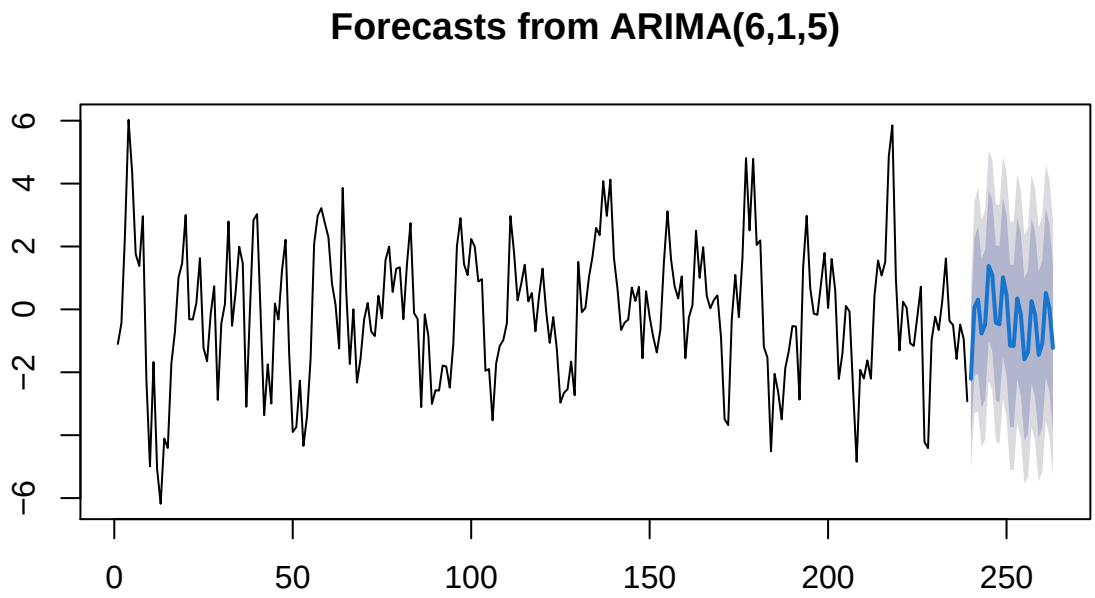


The first model we set up using the auto.arima() function is an ARIMA(p=3,d=0,q=1)(P=2,D=0,Q=1)[S=12] model with an AIC = -593.55. The second model is an ARIMA(6,1,5)(0,0,0)[0] model with an AIC of -529.33 and our final model is an ARIMA(3,0,5)(2,1,0)[12] model which had an AIC of -1.818842. Clearly, auto.arima() had the best choice according to the AIC. Therfore, our first model is my highest recommendation before performing cross validation.

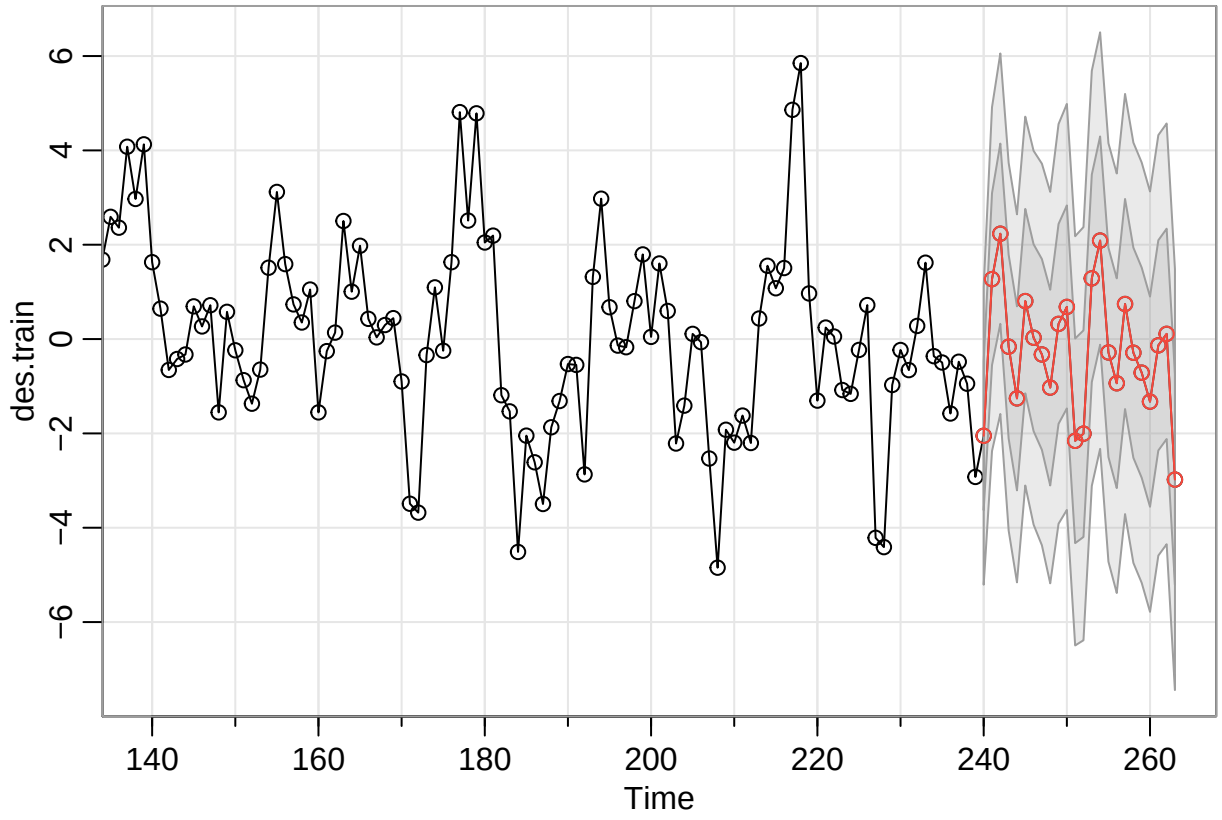
We can now perform some cross validation for forecasting and see how our models perform in a real test.

```
#Compare and cross validate
#Create training and test
set.seed(100)
deseasonal <- na.omit(decomp1$random)
des.train <- deseasonal[1:239]
des.test <- unlist(deseasonal[240:263])

fit1 <- auto.arima(des.train )
fcast1 <- forecast(fit1,h=24)

fit2 <- arima(des.train ,order = c(6,1,5))
fcast2 <- forecast(fit2,h=24)

par(mfrow = c(2,1))
plot(fcast1)
plot(fcast2)
```






```
fcast3 <- sarima.for(des.train,24 ,3,0,5,2,1,0,12)
```




```
mse1 <- mean(des.test-fcast1$mean)^2
mse1
```

```
## [1] 0.7063408
```

```
mse2 <- mean(des.test-fcast2$mean)^2
mse2
```

```
## [1] 0.9687369
```

```
mse3 <- mean(des.test-fcast3$pred)^2
mse3
```

```
## [1] 0.8026575
```

As we would have expected, the first model had the lowest MSE and seems to have performed the best out of all 3 models.I would recommend using this model for future analysis since it seems to have performed the best in a cross validation test as well as it having the lowest AIC. This gives us some ability to trust the auto.arima() function !