Predicting Fire Burn Occurrences

Lina Nguyen,Cole Bailey,Kevin Stewart

12/3/2021

Importing packages

```
library(tidyverse);library(tidyr);library(zoo);library(zoo);library(xts);libr
ary(tseries);library(astsa);library(lubridate);library(ggplot2);library(dplyr
);library(fpp2);library(dplyr);library(tsibble);library(tseries);library(fore
cast);library(sarima);library(fpp2);library(ggplot2)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse
1.3.1 ──

## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.3     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.3     ✓ stringr 1.4.0
## ✓ readr   2.0.0     ✓ forcats 0.5.1

## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()

##
## Attaching package: 'zoo'

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

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last

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

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

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

```
## ── Attaching packages ──────────────────────────────────────────── fpp2
2.4 ──
```

```
## ✓ forecast  8.15     ✓ expsmooth 2.3
## ✓ fma       2.4
```

```
##
```

```
##
## Attaching package: 'fpp2'
```

```
## The following object is masked from 'package:astsa':
##
##     oil
```

```
##
## Attaching package: 'tsibble'
```

```
## The following object is masked from 'package:lubridate':
##
##     interval
```

```
## The following object is masked from 'package:zoo':
##
##     index
```

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

```
## Loading required package: stats4
```

```
##
## Attaching package: 'sarima'
```

```
## The following object is masked from 'package:astsa':
##
##     sarima
```

#loading the data file and seperating string values

```
df_fire <-
read.csv("/Users/kevinstewart/Desktop/business_a/san_diego_fire_incident.csv"
, sep = ",")
```

#viewing data

```
head(df_fire)
```

```
##   fd_problem_nature_agg_datasd_v1            X
X.1
## 1                     agency_type address_city
problem
## 2                            Fire    SAN DIEGO                  Ringing
Alarm
## 3                            Fire    SAN DIEGO   Cardiac / Respiratory
Arrest
## 4                            Fire    SAN DIEGO                Assault/Rape
(L4)
## 5                            Fire    SAN DIEGO        Assist PD - Ladder
Bldg
## 6                            Fire    SAN DIEGO Back Pain (Non Traumatic)
(L4)
##             X.2            X.3           X.4
## 1 problem_count month_response year_response
## 2             1              1          1900
## 3             1              9          2001
## 4             2              9          2006
## 5             4              9          2006
## 6            42              9          2006
```

correct colnames & and subsetting data based on year and most occurring problem

```
names(df_fire) <- df_fire %>% slice(1) %>%
unlist()
fire <- df_fire %>% slice(-1)
```

Mutate the data to month response

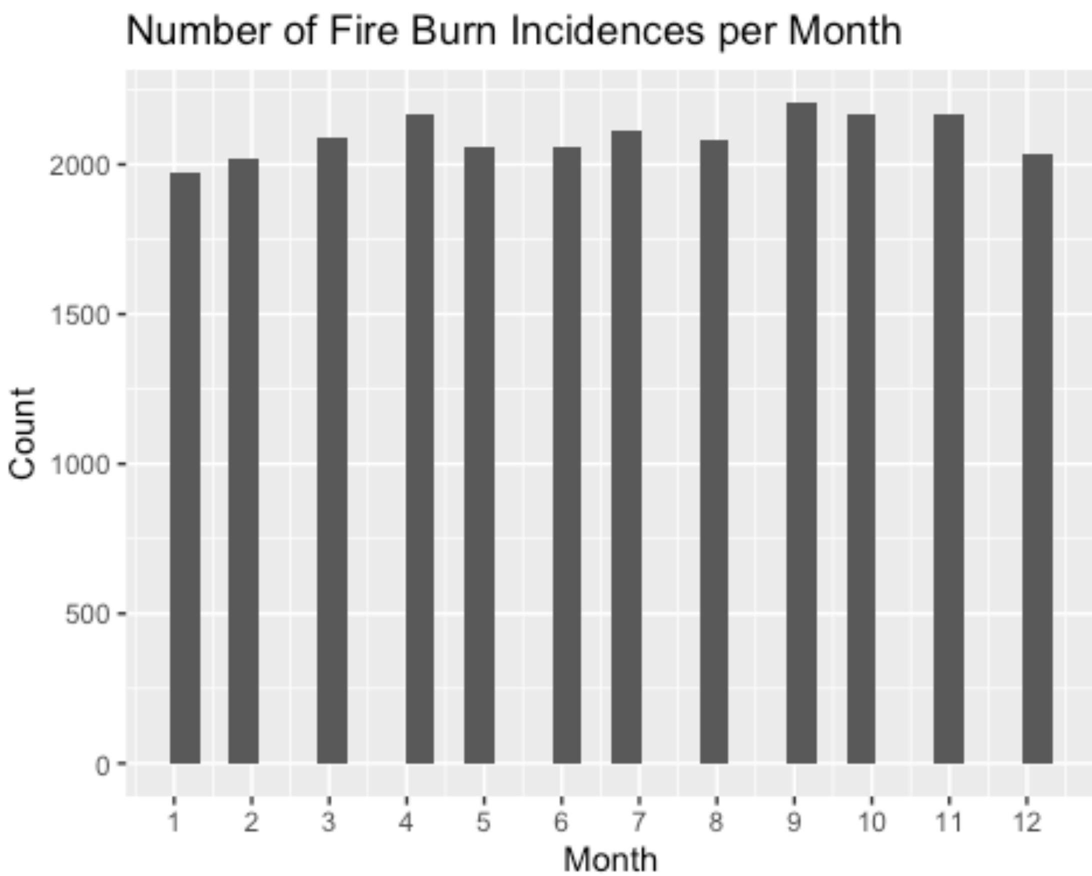
```
fire$problem_count <- as.numeric(fire$problem_count)
fire$month_response <- as.numeric(fire$month_response)
fire$year_response <- as.numeric(fire$year_response)
fire$problem <- as.factor(fire$problem)

########################################
# verify structure of data
str(fire)

## 'data.frame':    25138 obs. of  6 variables:
##  $ agency_type   : chr  "Fire" "Fire" "Fire" "Fire" ...
##  $ address_city  : chr  "SAN DIEGO" "SAN DIEGO" "SAN DIEGO" "SAN DIEGO"
...
##  $ problem       : Factor w/ 485 levels ".Confined Space/Trench
Rescue",..: 329 111 70 77 84 88 97 100 103 119 ...
##  $ problem_count : num  1 1 2 4 42 2 9 1 1 7 ...
##  $ month_response: num  1 9 9 9 9 9 9 9 9 9 ...
##  $ year_response : num  1900 2001 2006 2006 2006 ...
```

Verify the length and dimensions of the data

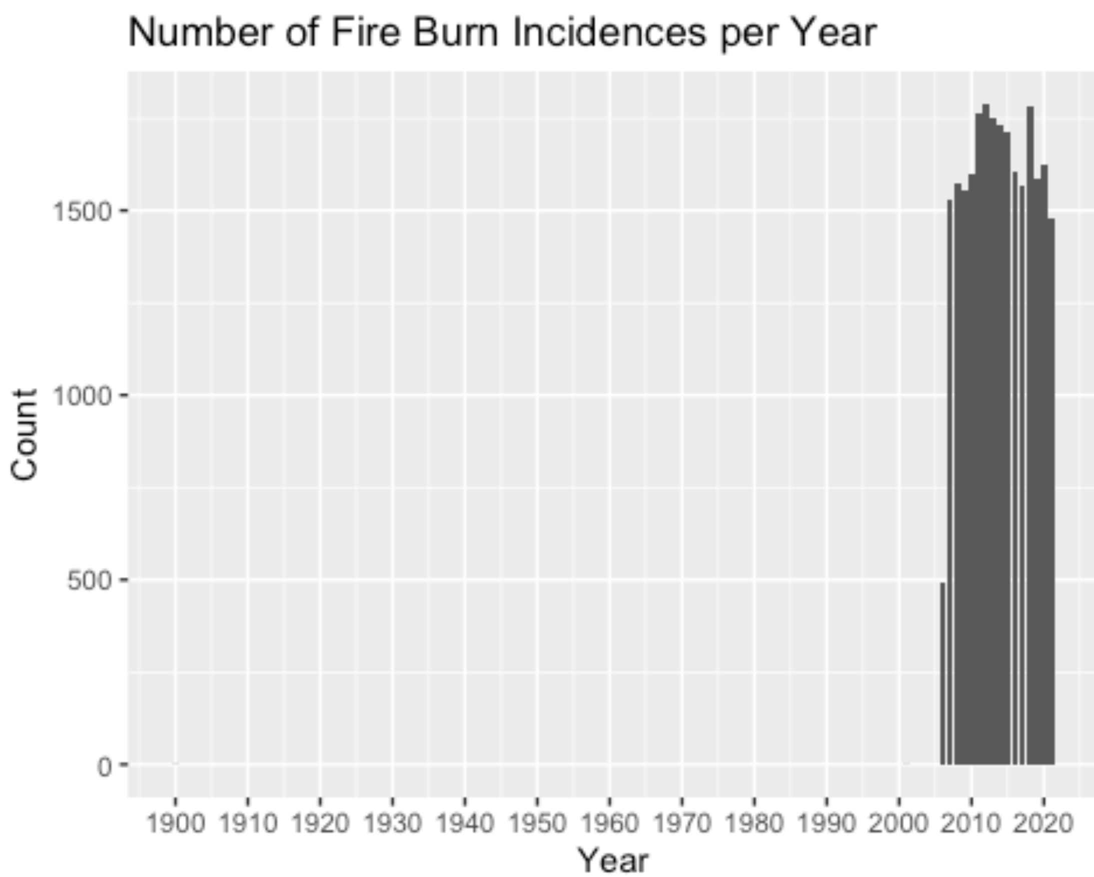
```
################################
# Instantiate the variable
burns_fire <- fire %>%
  filter(problem == "Burns / Explosion (L3)")
```

Viewing frequency of data

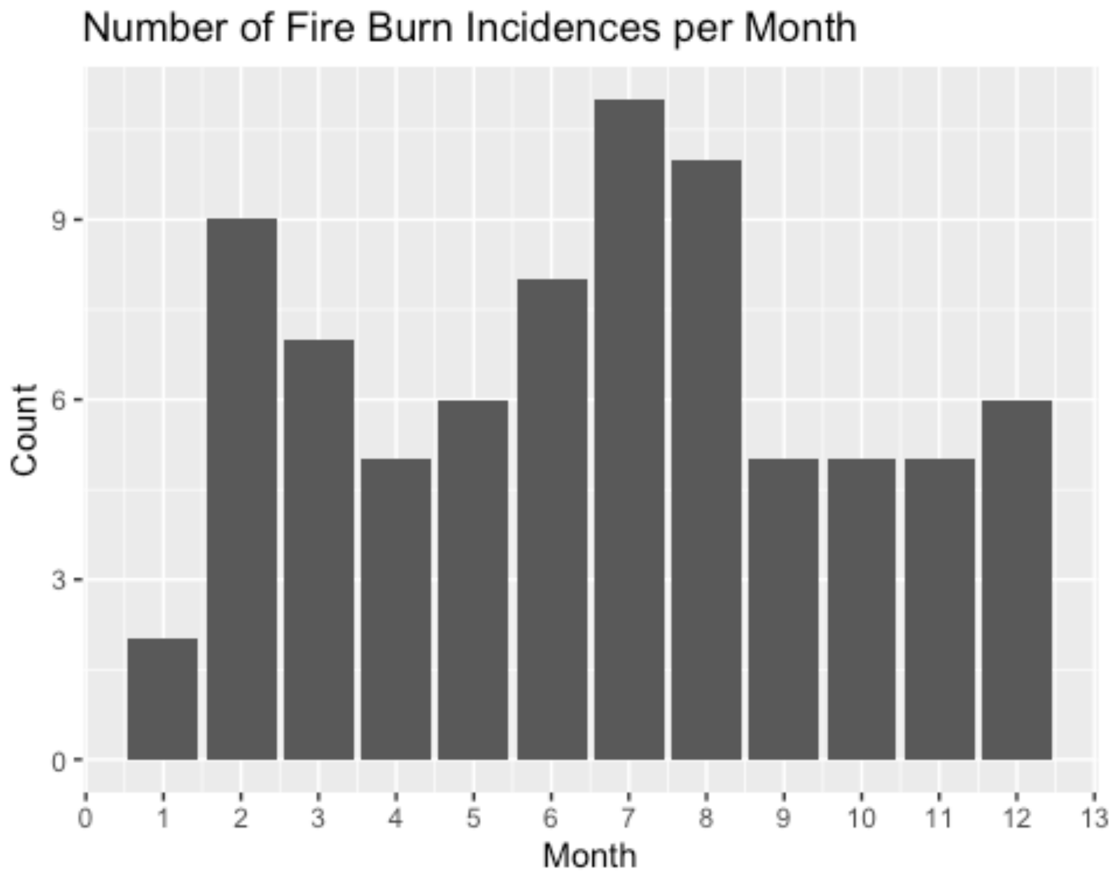
```
#############################################
ggplot(data = fire) + geom_histogram(mapping = aes(x = month_response)) +
scale_x_continuous(breaks = scales::pretty_breaks(n = 12)) + ggtitle('Number
of Fire Burn Incidences per Month') + xlab('Month') + ylab('Count')

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



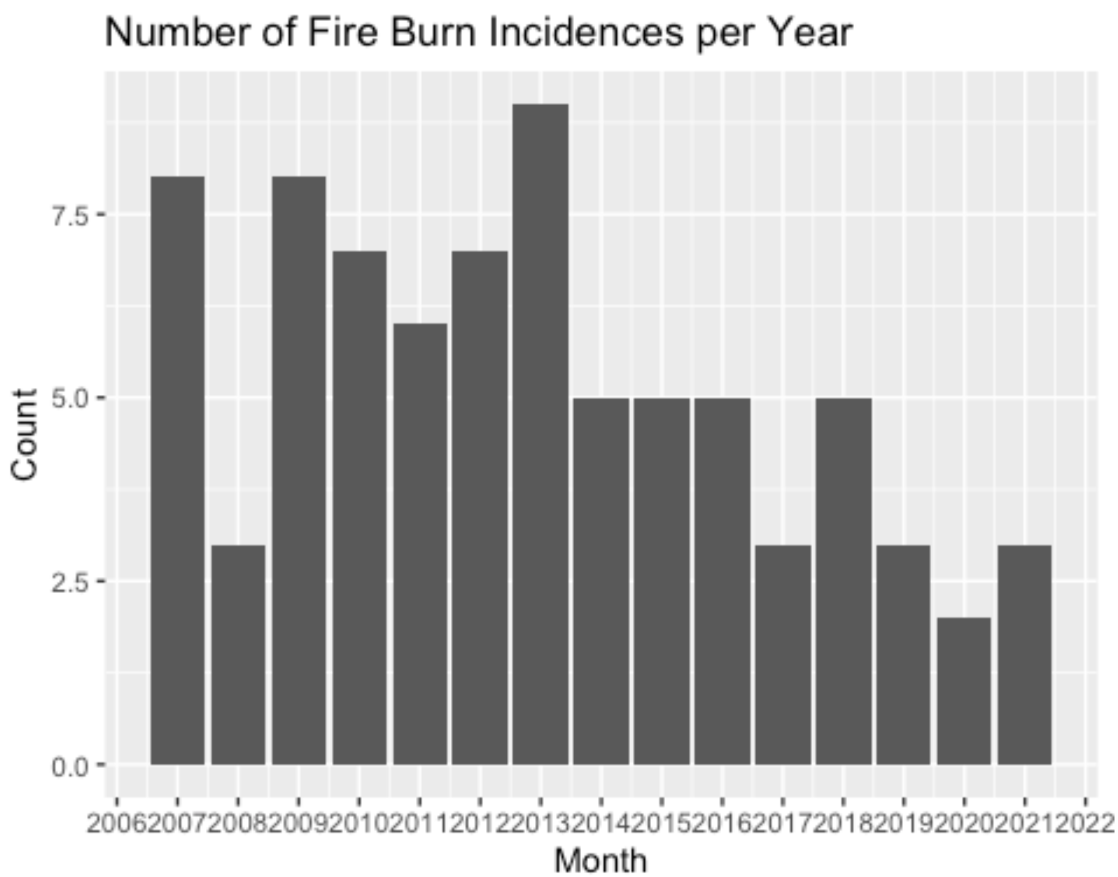

```
#############################################
ggplot(data = fire) + geom_bar(mapping = aes(x = year_response)) +
ggtitle("Number of Fire Burn Incidences per Year") + xlab("Year") +
ylab("Count") + scale_x_continuous(breaks = scales::pretty_breaks(n = 16))
```




```
###############################################
ggplot(data = burns_fire) + geom_bar(mapping = aes(x = month_response)) +
scale_x_continuous(breaks = scales::pretty_breaks(n = 12)) + ggtitle('Number
of Fire Burn Incidences per Month') + xlab('Month') + ylab('Count')
```

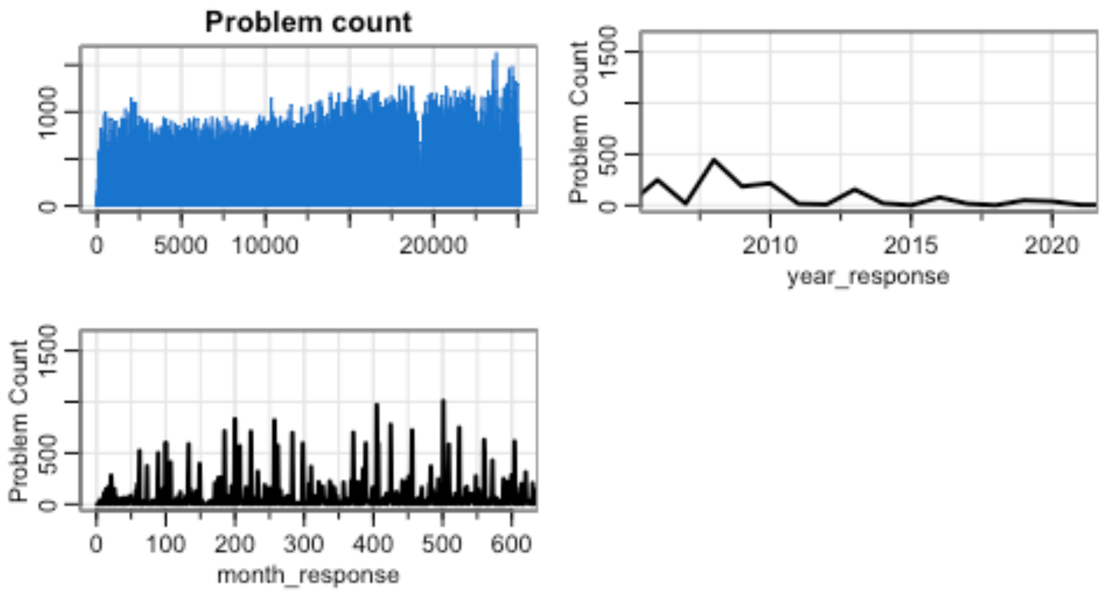



```
################################################
ggplot(data = burns_fire) + geom_bar(mapping = aes(x = year_response)) +
scale_x_continuous(breaks = scales::pretty_breaks(n = 12)) + ggtitle('Number
of Fire Burn Incidences per Year') + xlab('Month') + ylab('Count')
```



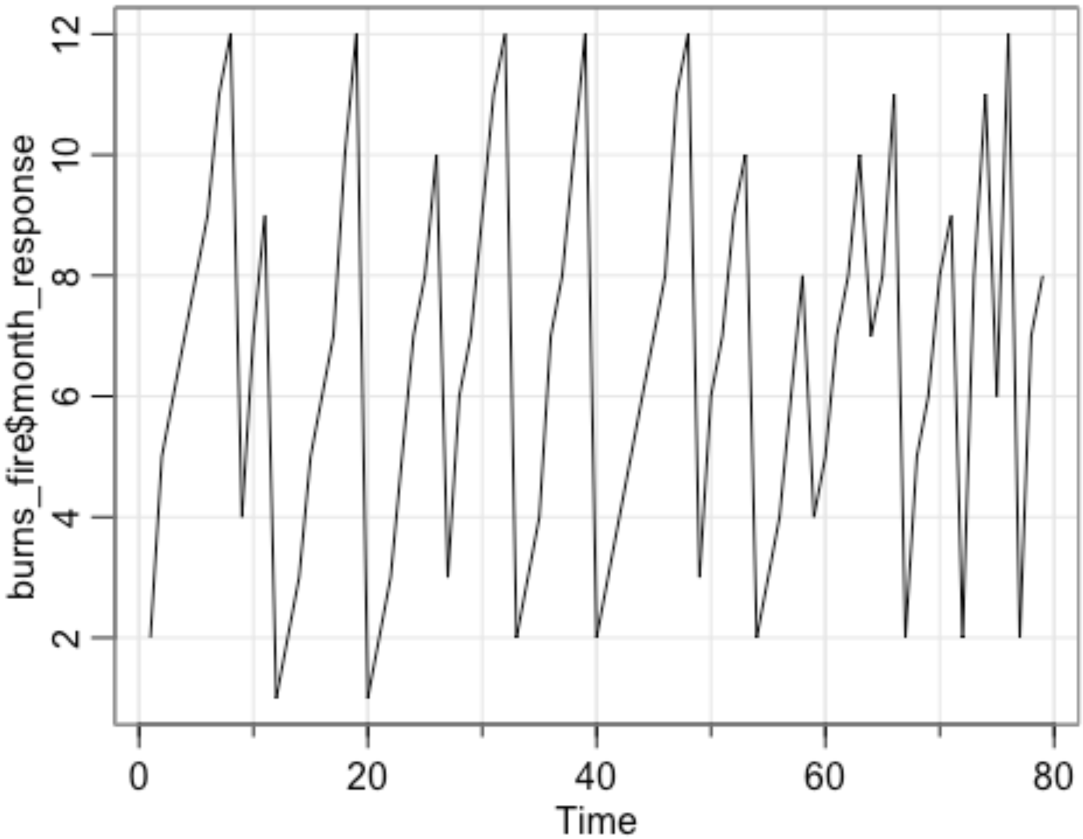
Looking at time series data

```
par(mfrow=c(3:2))
tsplot(fire$problem_count, ylab = "", xlab = "", type = "l", main = "Problem
count", col = 4)
tsplot(fire[,4], xlab = "year_response", ylab = "Problem Count", type = "l",
lwd=2, xlim = c(2006, 2021))
tsplot(fire[,4], xlab = "month_response", ylab = "Problem Count", type = "l",
lwd=2, xlim = c(1,612))
```

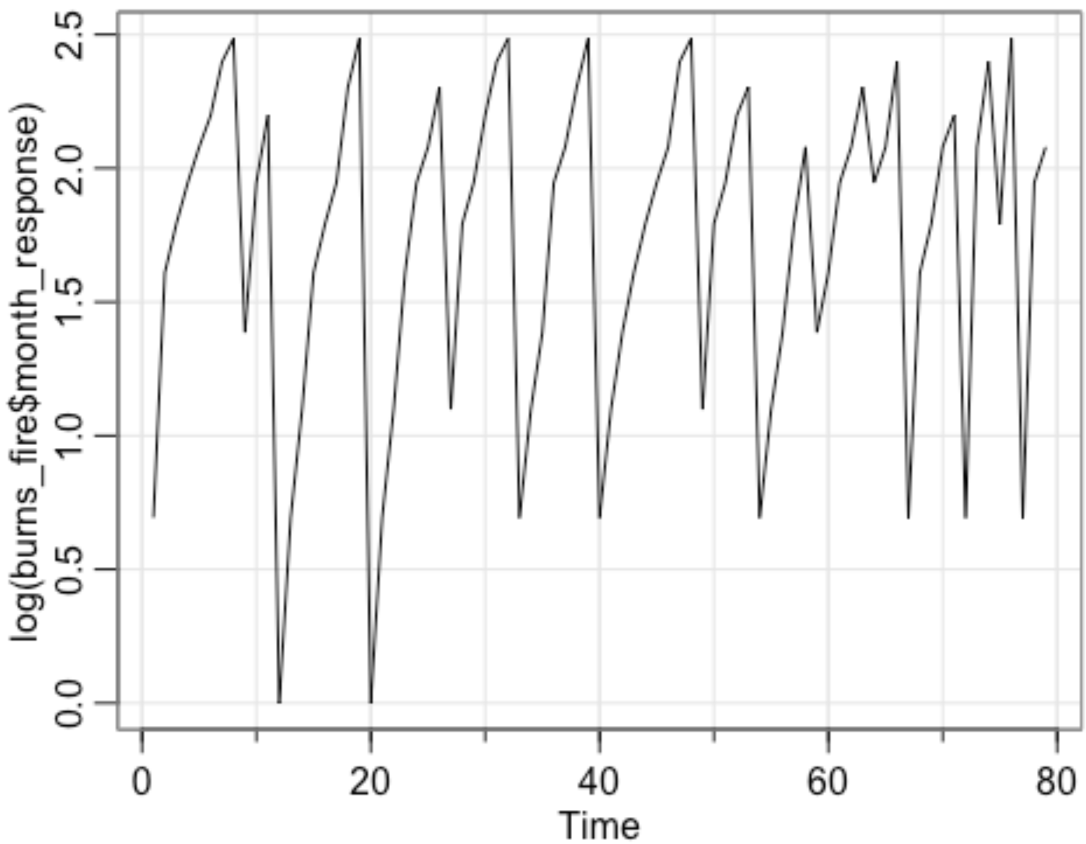


view data by individual fire incidents

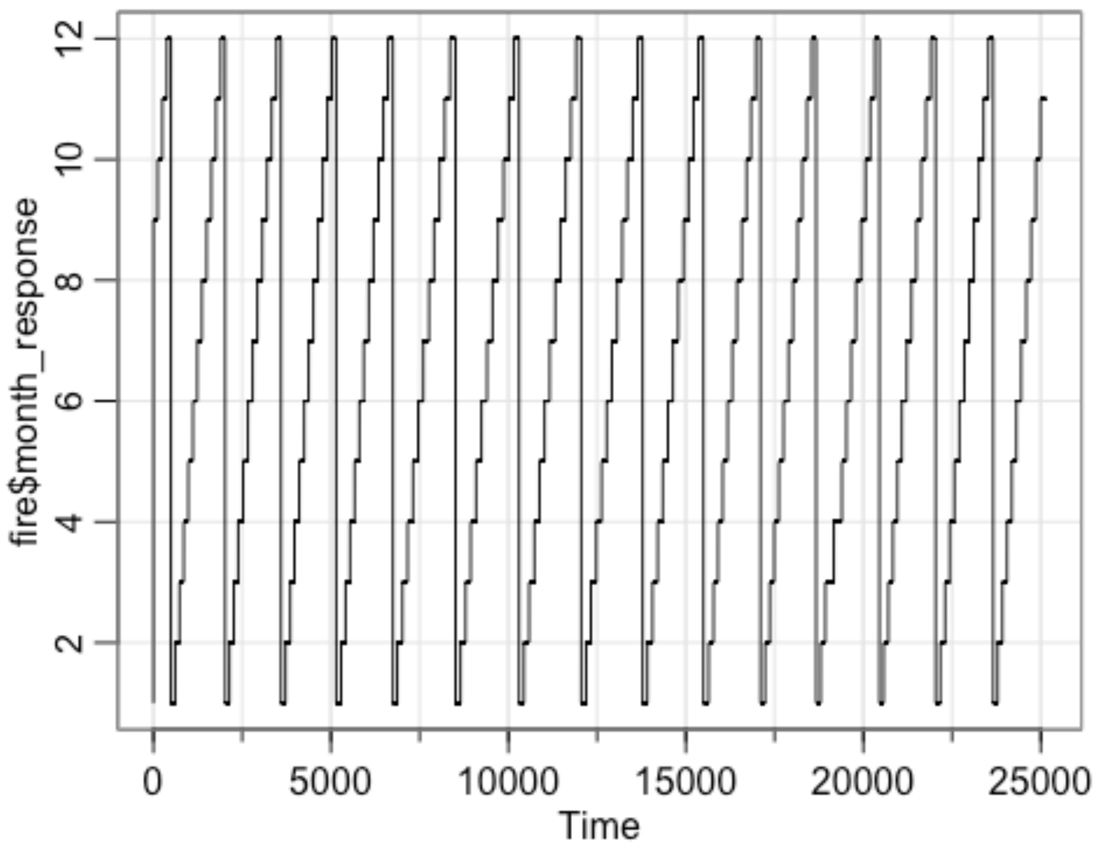
```
tsplot(burns_fire$month_response)
```



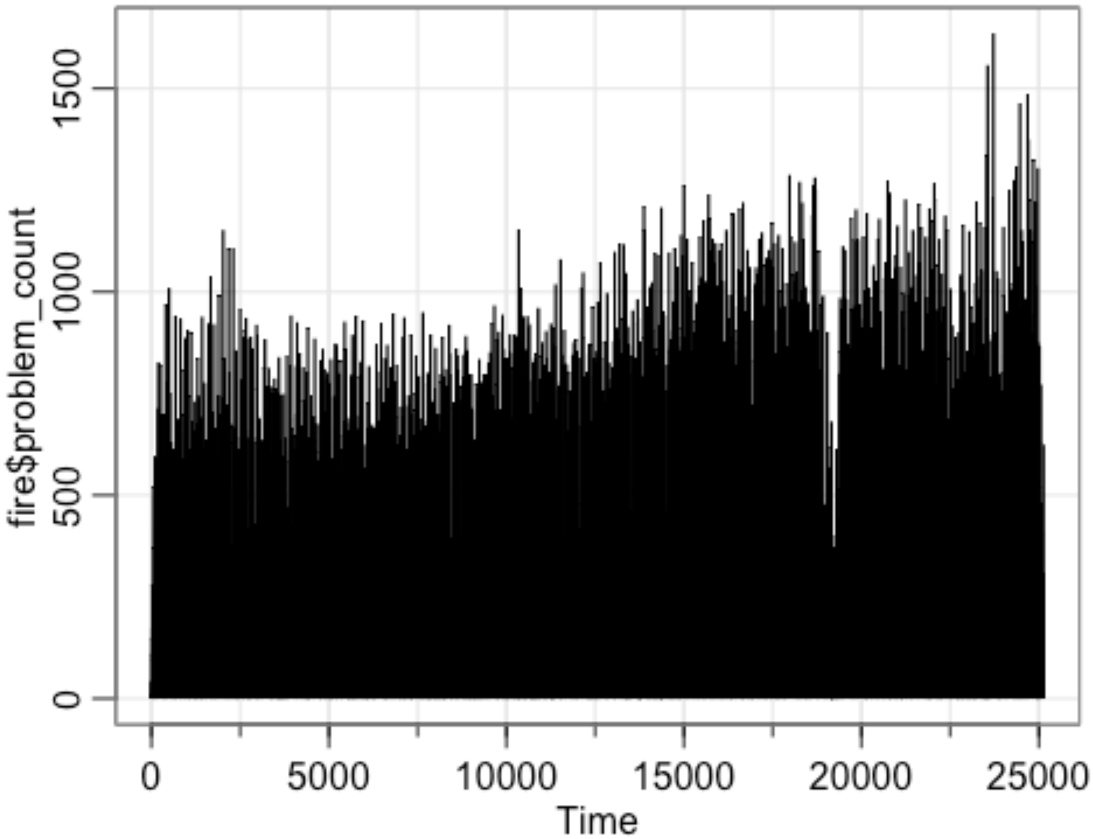

```
# logging the data graphically
tsplot(log(burns_fire$month_response))
```




```
# the data shows that there is a there is also seasonality in the data which
will need to be removed.
########################################
#viewing first variables of the burns month response data
head(burns_fire$month_response)

## [1] 2 5 6 7 8 9

#####################################
# graphically displaying month response and problem count
tsplot(fire$month_response)
```

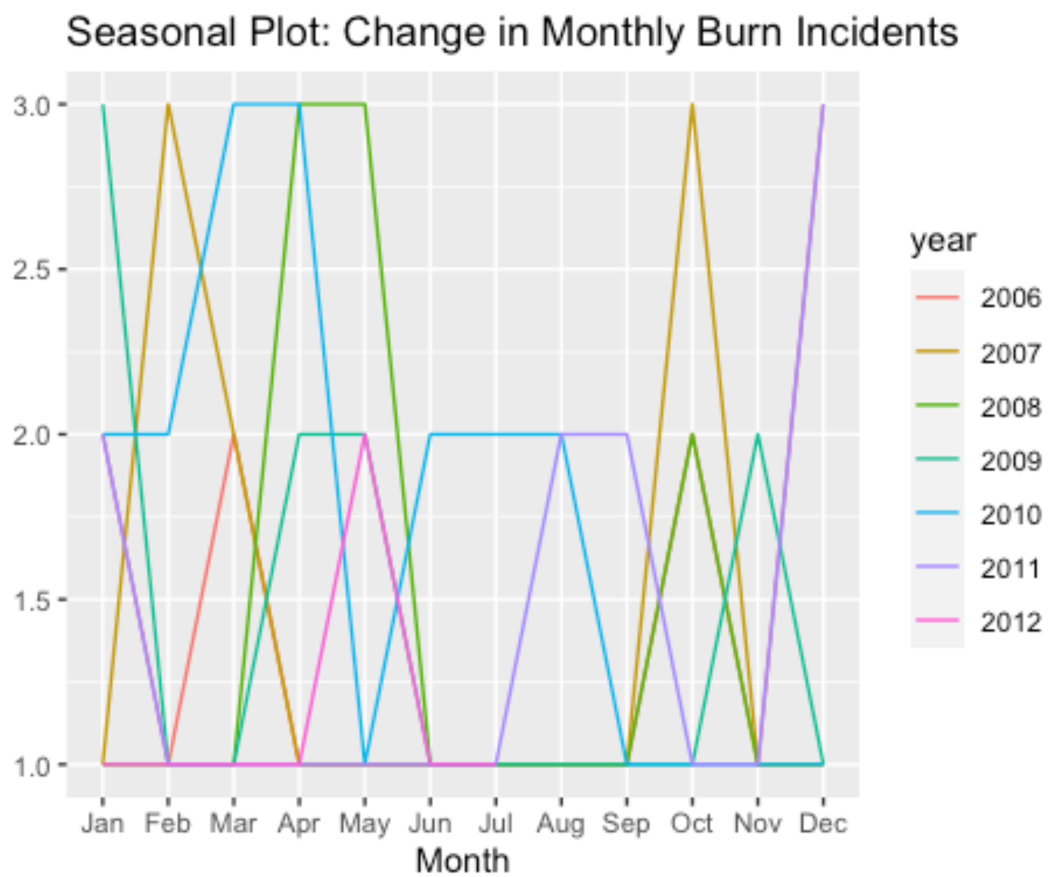



```
tsplot(fire$problem_count)
```

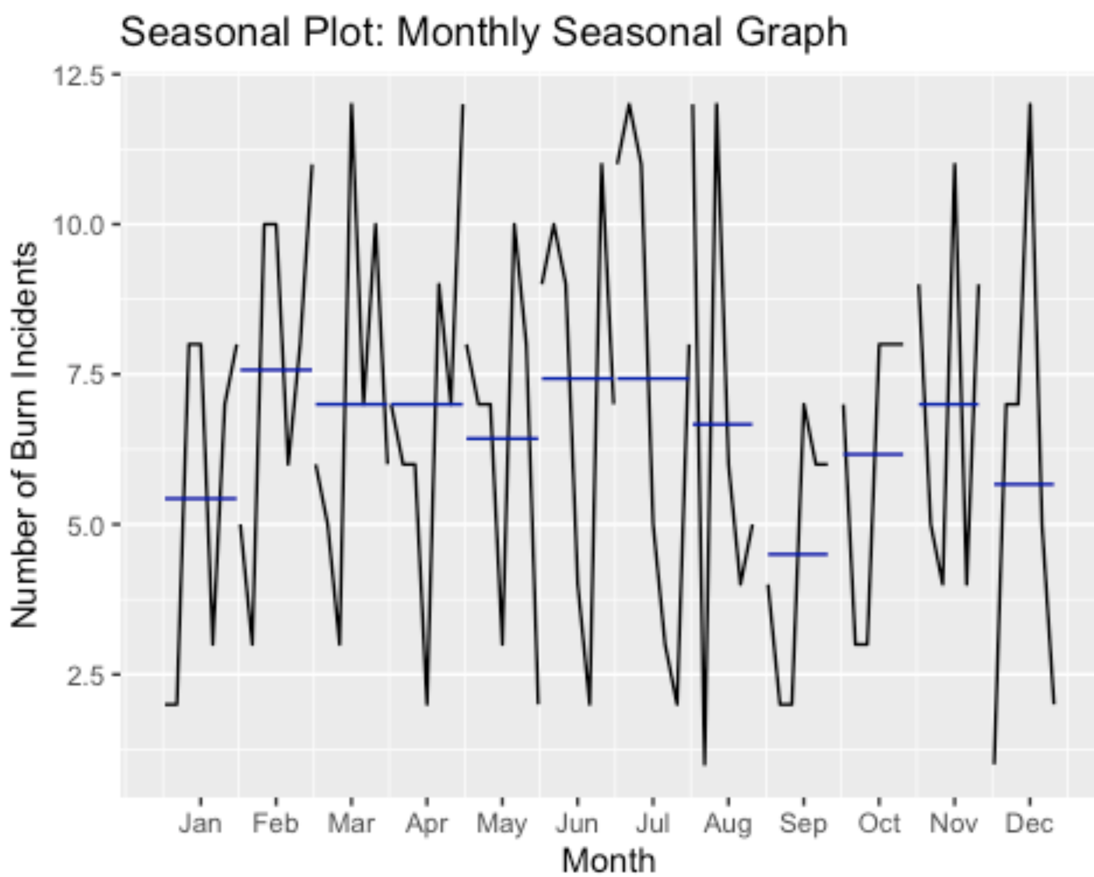


Declaring data as time series

```
# creating a time series variable
Y = ts(burns_fire[,4],start = c(2006,1),frequency = 12)
###############################################
# create a time series object to check for seasonality
ts_fire_b <- ts(burns_fire[,5],start = c(2006,1),frequency = 12)

# Checking seasonality with the seasonal graph
ggseasonplot(Y) + ggtitle("Seasonal Plot: Change in Monthly Burn Incidents ")
```



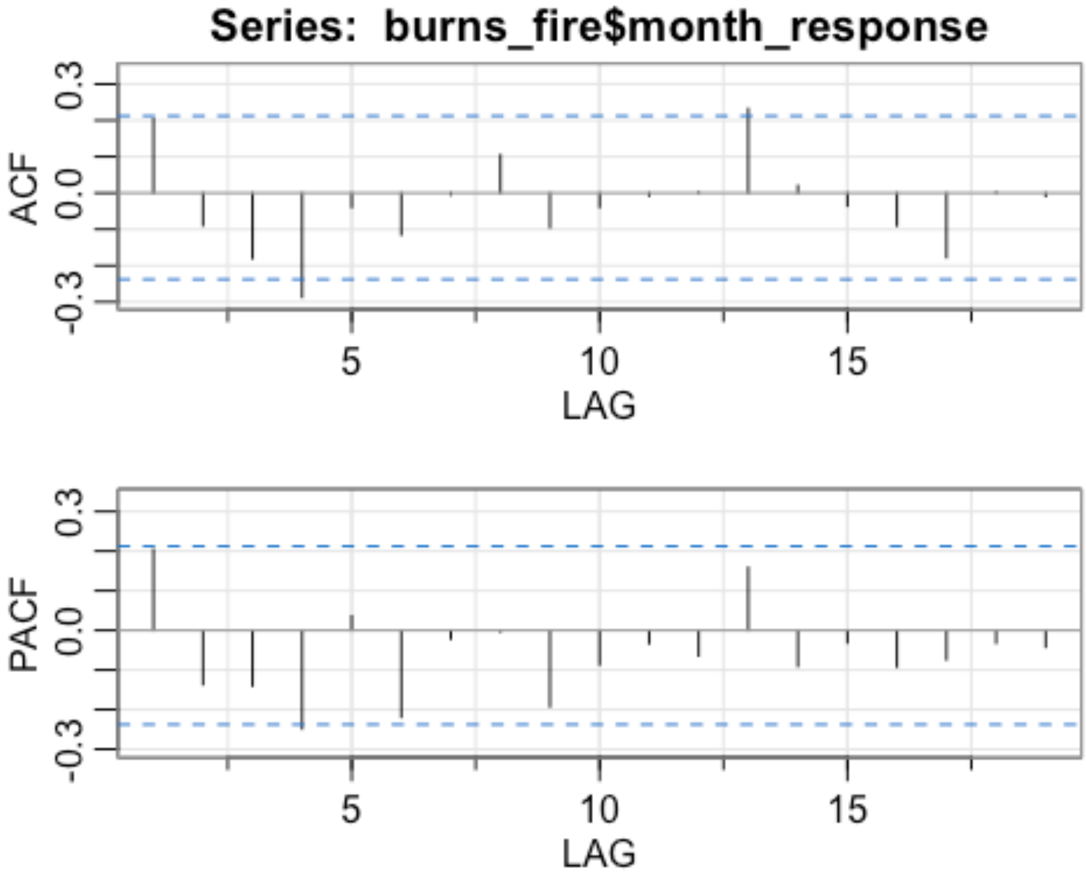

```
###############################################
ggsubseriesplot(ts_fire_b) +
  ggtitle("Seasonal Plot: Monthly Seasonal Graph")+ ylab("Number of Burn
Incidents")
```




```
#There shows that there is a spike in the winter to spring months and drastic decrease at the start of summer.
```

Looking at acf and pacf plot to check autocorrelation

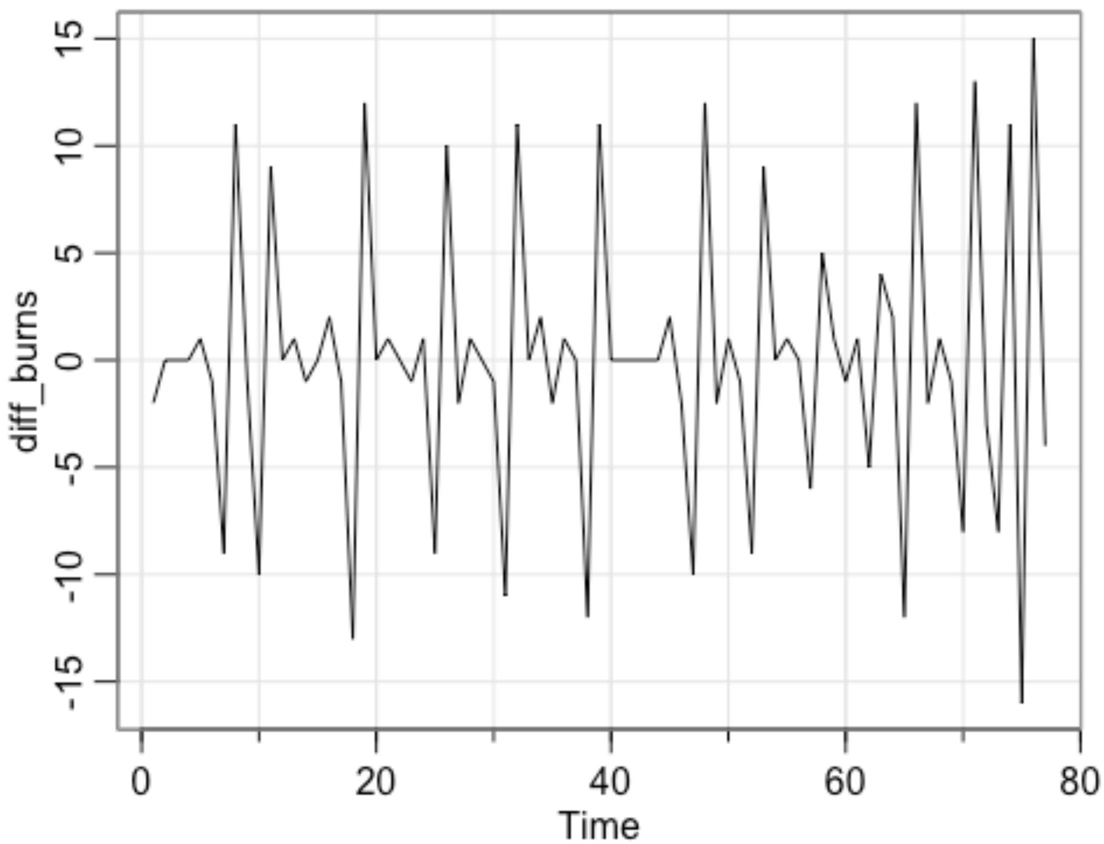
```
head(acf2(burns_fire$month_response))
```



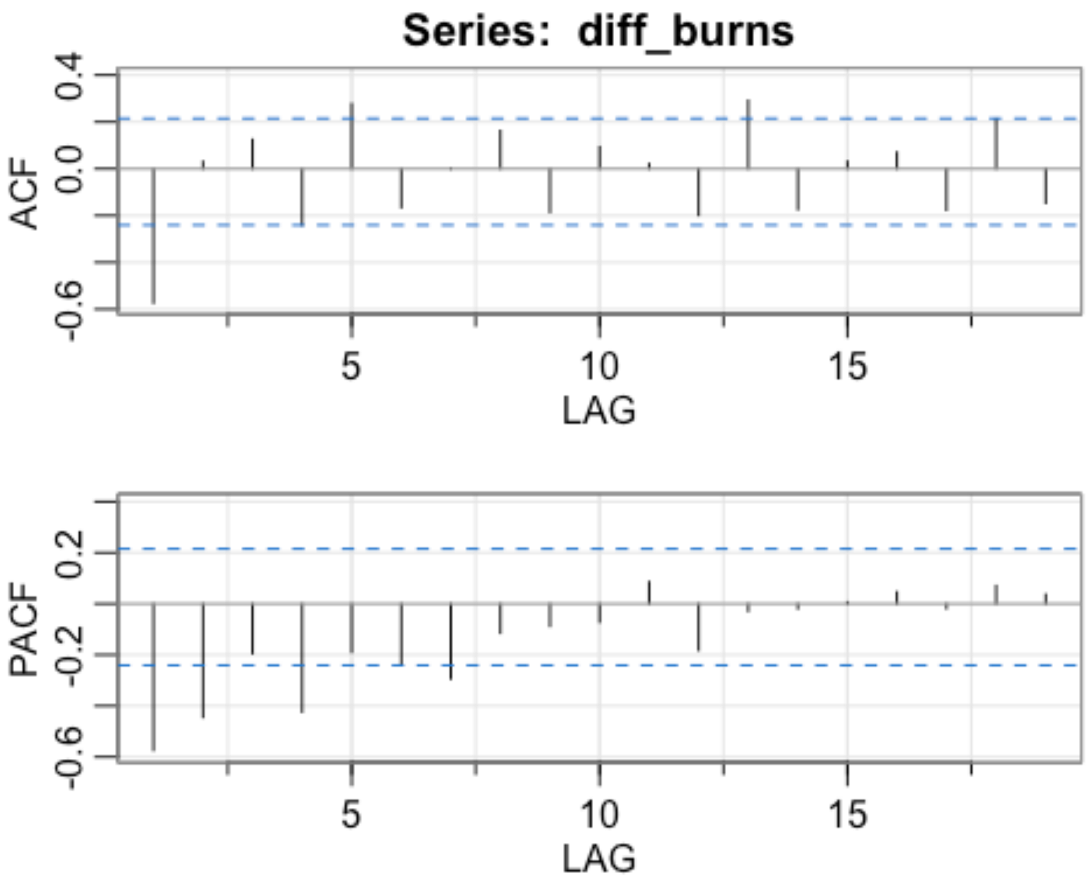

```
##      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7] [,8]  [,9] [,10] [,11] [,12]
## ACF   0.2 -0.09 -0.18 -0.29 -0.04 -0.12  0.00  0.1 -0.09 -0.04 -0.01  0.00
## PACF  0.2 -0.14 -0.14 -0.25  0.04 -0.22 -0.02  0.0 -0.19 -0.09 -0.03 -0.06
##      [,13] [,14] [,15] [,16] [,17] [,18] [,19]
## ACF   0.23  0.02 -0.04 -0.09 -0.18  0.00 -0.01
## PACF  0.16 -0.09 -0.03 -0.09 -0.07 -0.03 -0.04

#this show a significant amount of correlation in
```

#Differencing the data to obtain stationarity to remove trend and seasonality

```
diff_burns <- diff(diff(burns_fire$month_response))
tsplot(diff_burns)
```




```
# acf and pacf plots
acf2(diff_burns)
```




```
##       [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10] [,11] 
[,12]
## ACF  -0.57  0.03  0.12 -0.24  0.28 -0.17  0.00  0.16 -0.19  0.09  0.02 -
0.20
## PACF -0.57 -0.44 -0.19 -0.42 -0.19 -0.24 -0.29 -0.11 -0.09 -0.07  0.09 -
0.18
##      [,13] [,14] [,15] [,16] [,17] [,18] [,19]
## ACF   0.29 -0.17  0.03  0.07 -0.18  0.21 -0.15
## PACF -0.03 -0.02  0.01  0.05 -0.02  0.07  0.04
```

Trying different AR models to find the best model

```
# Looking at ARIMA (1,1,1) model
ar1 <- arima(ts_fire_b, order = c(1,1,1))
ar1

## 
## Call:
## arima(x = ts_fire_b, order = c(1, 1, 1))
## 
## Coefficients:
##          ar1      ma1
##       0.2231  -1.0000
```

```
## s.e.  0.1128   0.0378
##
## sigma^2 estimated as 9.518:  log likelihood = -200.51,  aic = 407.03

# AIC of 262.73
###################################
# ARIMA (2,0,0)
ar2 <- arima(ts_fire_b, order = c(2,0,0))
ar2

##
## Call:
## arima(x = ts_fire_b, order = c(2, 0, 0))
##
## Coefficients:
##          ar1      ar2  intercept
##       0.2350  -0.1365     6.5549
## s.e.  0.1123   0.1117     0.3792
##
## sigma^2 estimated as 9.218:  log likelihood = -199.87,  aic = 407.74

###################################
# In viewing the data it appears that the data is best with differncing
ar3 <- arima(ts_fire_b, order = c(3,0,0))
ar3

##
## Call:
## arima(x = ts_fire_b, order = c(3, 0, 0))
##
## Coefficients:
##          ar1      ar2      ar3  intercept
##       0.2153  -0.1030  -0.1446     6.5708
## s.e.  0.1121   0.1134   0.1125     0.3293
##
## sigma^2 estimated as 9.022:  log likelihood = -199.05,  aic = 408.11

###################################
ar4 <- arima(ts_fire_b, order =c(4,0,0))
ar4

##
## Call:
## arima(x = ts_fire_b, order = c(4, 0, 0))
##
## Coefficients:
##          ar1      ar2      ar3      ar4  intercept
##       0.1752  -0.1322  -0.0793  -0.2588     6.5735
## s.e.  0.1097   0.1110   0.1123   0.1125     0.2558
##
## sigma^2 estimated as 8.432:  log likelihood = -196.51,  aic = 405.03
```

```
####################################
ar5 <- arima(ts_fire_b, order = c(5,0,0))
ar5

##
## Call:
## arima(x = ts_fire_b, order = c(5, 0, 0))
##
## Coefficients:
##          ar1      ar2      ar3      ar4     ar5  intercept
##       0.1876  -0.1275  -0.0715  -0.2706  0.0498     6.5720
## s.e.  0.1132   0.1114   0.1136   0.1156  0.1162     0.2681
##
## sigma^2 estimated as 8.41:  log likelihood = -196.42,  aic = 406.85

print(ar1);ar2;ar3;ar4;ar5

##
## Call:
## arima(x = ts_fire_b, order = c(1, 1, 1))
##
## Coefficients:
##          ar1      ma1
##       0.2231  -1.0000
## s.e.  0.1128   0.0378
##
## sigma^2 estimated as 9.518:  log likelihood = -200.51,  aic = 407.03

##
## Call:
## arima(x = ts_fire_b, order = c(2, 0, 0))
##
## Coefficients:
##          ar1      ar2  intercept
##       0.2350  -0.1365     6.5549
## s.e.  0.1123   0.1117     0.3792
##
## sigma^2 estimated as 9.218:  log likelihood = -199.87,  aic = 407.74

##
## Call:
## arima(x = ts_fire_b, order = c(3, 0, 0))
##
## Coefficients:
##          ar1      ar2      ar3  intercept
##       0.2153  -0.1030  -0.1446     6.5708
## s.e.  0.1121   0.1134   0.1125     0.3293
##
## sigma^2 estimated as 9.022:  log likelihood = -199.05,  aic = 408.11
```

```
##
## Call:
## arima(x = ts_fire_b, order = c(4, 0, 0))
##
## Coefficients:
##          ar1      ar2      ar3      ar4  intercept
##       0.1752  -0.1322  -0.0793  -0.2588     6.5735
## s.e.  0.1097   0.1110   0.1123   0.1125     0.2558
##
## sigma^2 estimated as 8.432:  log likelihood = -196.51,  aic = 405.03
```

```
##
## Call:
## arima(x = ts_fire_b, order = c(5, 0, 0))
##
## Coefficients:
##          ar1      ar2      ar3      ar4     ar5  intercept
##       0.1876  -0.1275  -0.0715  -0.2706  0.0498     6.5720
## s.e.  0.1132   0.1114   0.1136   0.1156  0.1162     0.2681
##
## sigma^2 estimated as 8.41:  log likelihood = -196.42,  aic = 406.85
```

```
####################################
# It is determined that an ARIMA model of (4,0,0) is the best model for
forecasting
```

Find the best arima model using a fit arima

```
# Determining best model
fit_ar <- auto.arima(diff_burns, stepwise = FALSE, approximation = FALSE,
trace = TRUE)
```

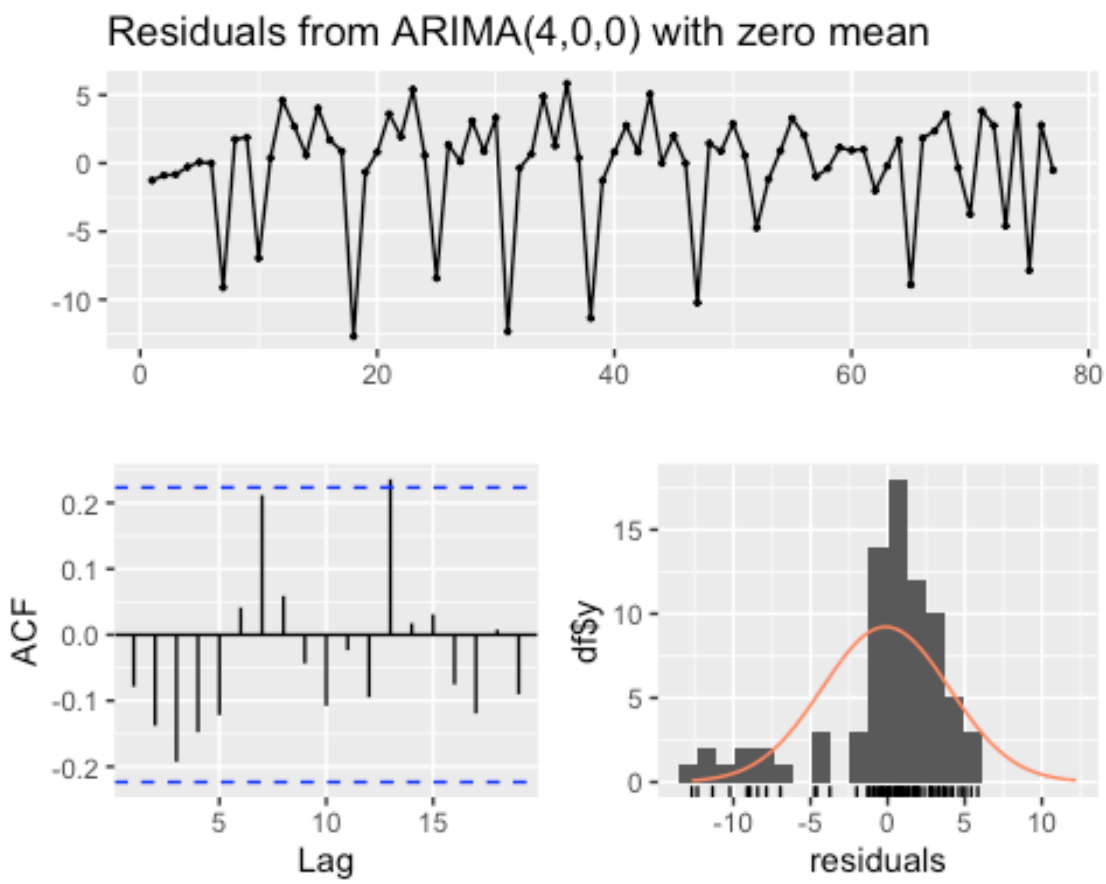
```
##
##  ARIMA(0,0,0) with zero mean     : 506.6855
##  ARIMA(0,0,0) with non-zero mean : 508.793
##  ARIMA(0,0,1) with zero mean     : Inf
##  ARIMA(0,0,1) with non-zero mean : Inf
##  ARIMA(0,0,2) with zero mean     : Inf
##  ARIMA(0,0,2) with non-zero mean : Inf
##  ARIMA(0,0,3) with zero mean     : Inf
##  ARIMA(0,0,3) with non-zero mean : Inf
##  ARIMA(0,0,4) with zero mean     : Inf
##  ARIMA(0,0,4) with non-zero mean : Inf
##  ARIMA(0,0,5) with zero mean     : Inf
##  ARIMA(0,0,5) with non-zero mean : Inf
##  ARIMA(1,0,0) with zero mean     : 478.2234
##  ARIMA(1,0,0) with non-zero mean : 480.3899
##  ARIMA(1,0,1) with zero mean     : Inf
##  ARIMA(1,0,1) with non-zero mean : Inf
##  ARIMA(1,0,2) with zero mean     : Inf
```

```
##  ARIMA(1,0,2) with non-zero mean : Inf
##  ARIMA(1,0,3) with zero mean     : Inf
##  ARIMA(1,0,3) with non-zero mean : Inf
##  ARIMA(1,0,4) with zero mean     : Inf
##  ARIMA(1,0,4) with non-zero mean : Inf
##  ARIMA(2,0,0) with zero mean     : 463.2976
##  ARIMA(2,0,0) with non-zero mean : 465.516
##  ARIMA(2,0,1) with zero mean     : Inf
##  ARIMA(2,0,1) with non-zero mean : Inf
##  ARIMA(2,0,2) with zero mean     : Inf
##  ARIMA(2,0,2) with non-zero mean : Inf
##  ARIMA(2,0,3) with zero mean     : Inf
##  ARIMA(2,0,3) with non-zero mean : Inf
##  ARIMA(3,0,0) with zero mean     : 462.9781
##  ARIMA(3,0,0) with non-zero mean : 465.2571
##  ARIMA(3,0,1) with zero mean     : Inf
##  ARIMA(3,0,1) with non-zero mean : Inf
##  ARIMA(3,0,2) with zero mean     : Inf
##  ARIMA(3,0,2) with non-zero mean : Inf
##  ARIMA(4,0,0) with zero mean     : 448.2129
##  ARIMA(4,0,0) with non-zero mean : 450.521
##  ARIMA(4,0,1) with zero mean     : Inf
##  ARIMA(4,0,1) with non-zero mean : Inf
##  ARIMA(5,0,0) with zero mean     : 448.8007
##  ARIMA(5,0,0) with non-zero mean : 451.166
##
##
##
##  Best model: ARIMA(4,0,0) with zero mean
```

```
###################################
# Confirming that an ARIMA (4,0,0) is the best fit model
```

#check residuals for the best model

```
#check residuals using forecast and a lag
forecast::checkresiduals(fit_ar, lag=12)
```

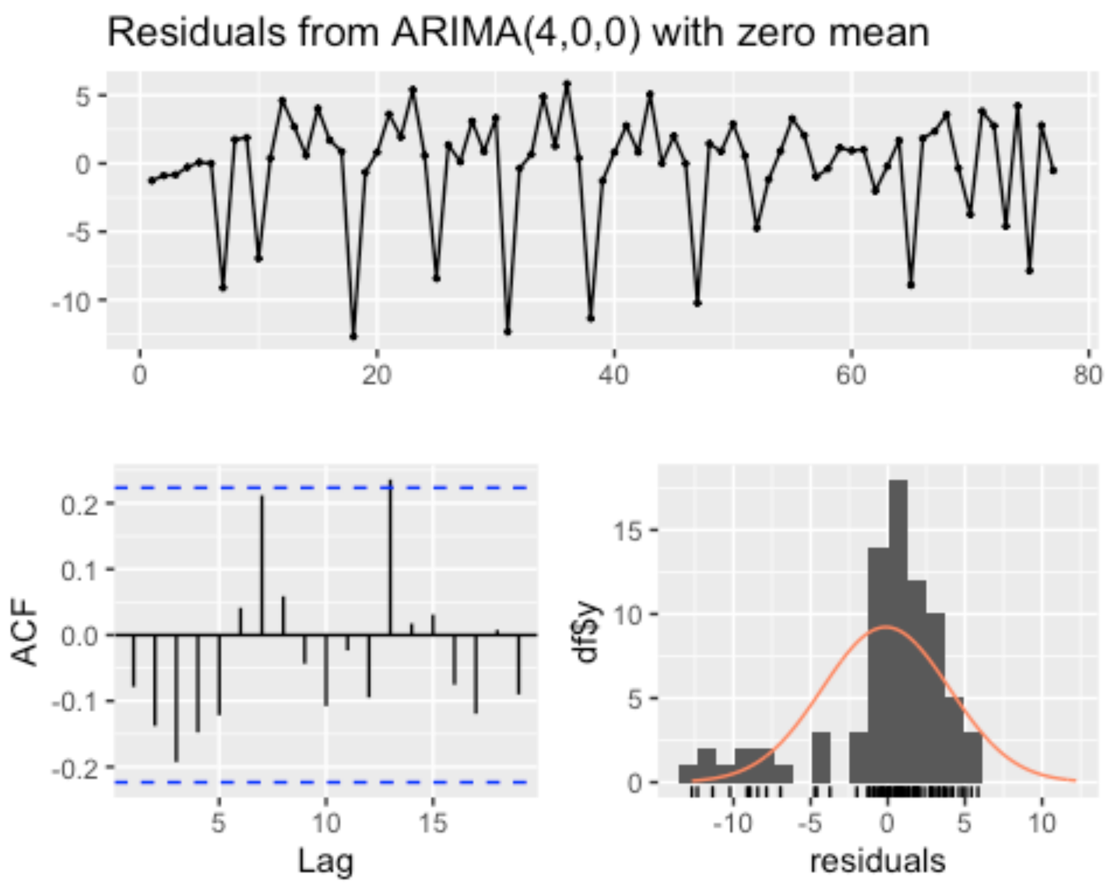



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,0,0) with zero mean
## Q* = 14.637, df = 8, p-value = 0.06659
##
## Model df: 4.   Total lags used: 12
```

```
# the best model based on forecasting and checking residuals
arima(burns_fire$month_response, order = c(4,0,0))
```

```
##
## Call:
## arima(x = burns_fire$month_response, order = c(4, 0, 0))
##
## Coefficients:
##          ar1      ar2      ar3      ar4  intercept
##       0.1752  -0.1322  -0.0793  -0.2588     6.5735
## s.e.  0.1097   0.1110   0.1123   0.1125     0.2558
##
## sigma^2 estimated as 8.432:  log likelihood = -196.51,  aic = 405.03
```

```
# print summary statistics
checkresiduals(fit_ar)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,0,0) with zero mean
## Q* = 13.746, df = 6, p-value = 0.03261
##
## Model df: 4.   Total lags used: 10

print(summary(fit_ar))

## Series: diff_burns
## ARIMA(4,0,0) with zero mean
##
## Coefficients:
##           ar1      ar2      ar3      ar4
##       -0.9945  -0.8816  -0.5776  -0.4603
## s.e.   0.1009   0.1372   0.1354   0.1035
##
## sigma^2 estimated as 17.64:  log likelihood=-218.68
## AIC=447.37   AICc=448.21   BIC=459.09
##
## Training set error measures:
```

```
##                      ME     RMSE      MAE MPE MAPE      MASE        ACF1
## Training set -0.1268859 4.089907 2.786639 NaN  Inf 0.3383141 -0.07872935

#Get the standard deviation
std <- sqrt(17.64)
std

## [1] 4.2
```

In viewing the acf most of all the autocorrelation is removed from the model.

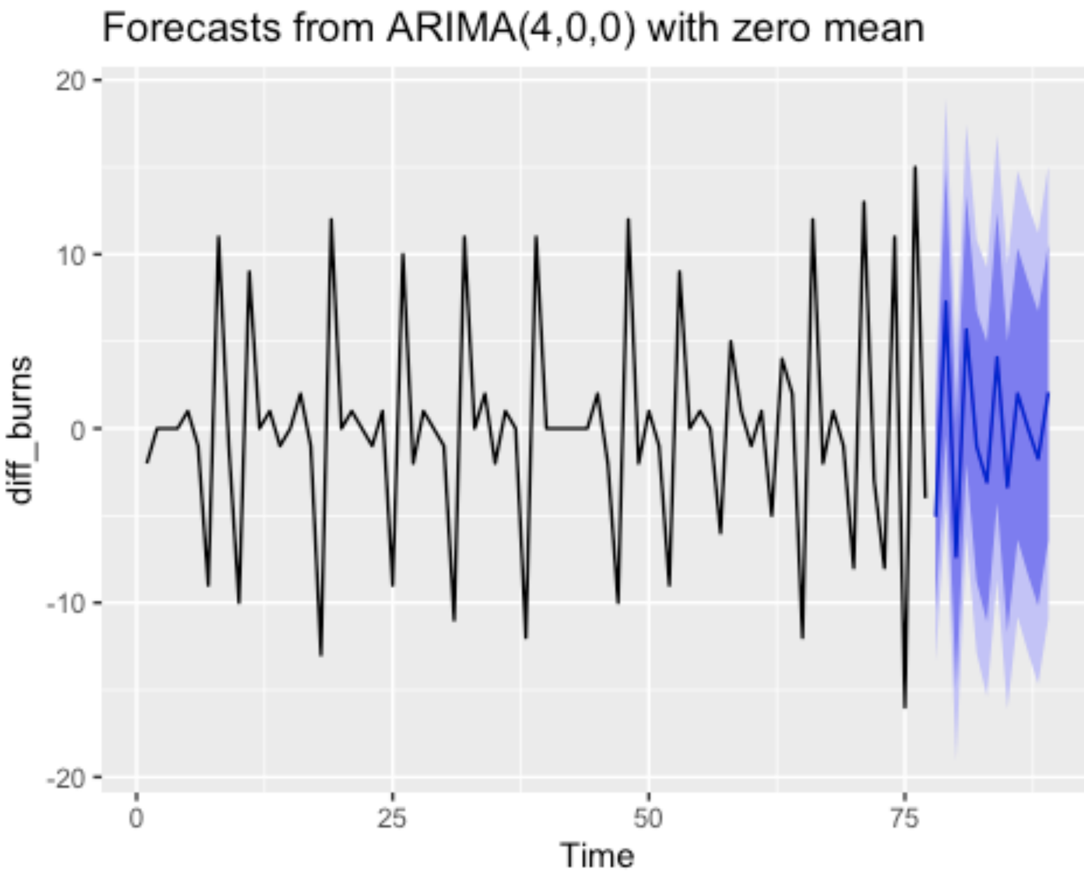
Dickey fuller Test to check for stationarity

```
adf.test(diff_burns)

## Warning in adf.test(diff_burns): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  diff_burns
## Dickey-Fuller = -7.0975, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary

#The stationarity is removed according to the dickey fuller test
```

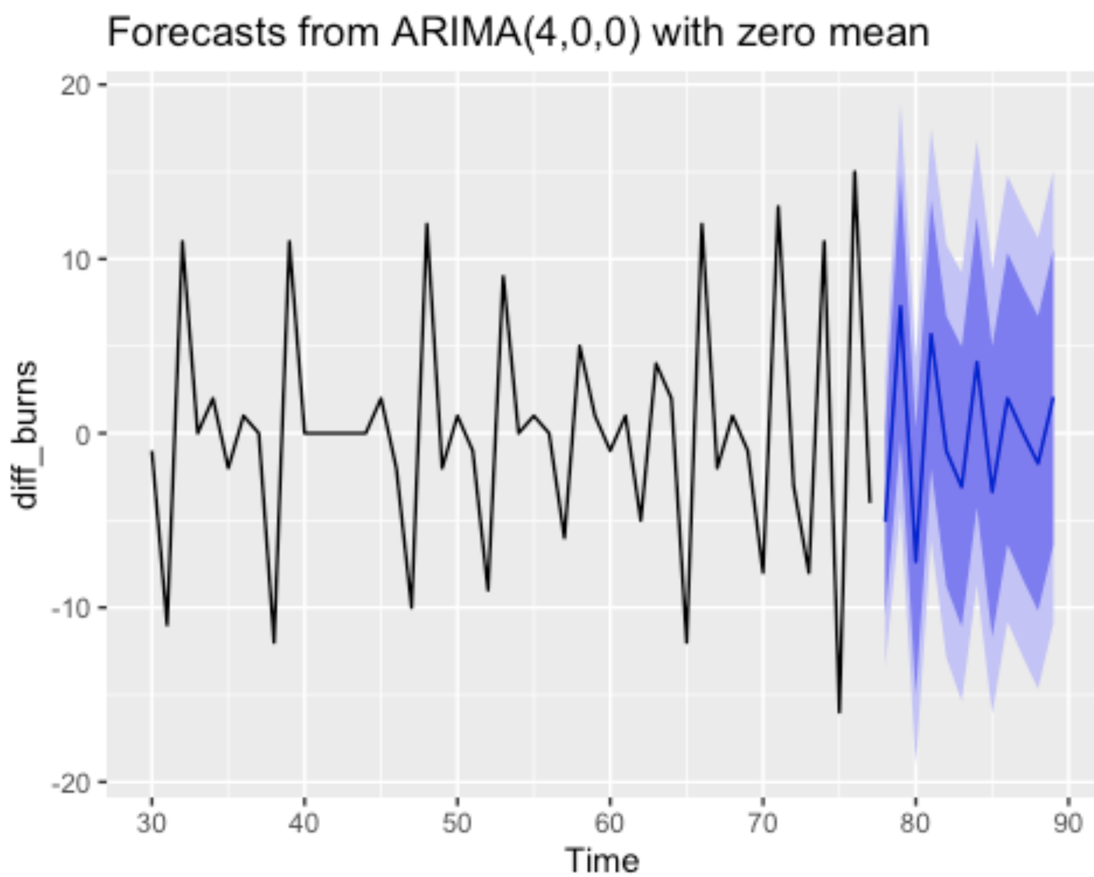
We would reject the null, shows that we have stationarity.

Using arima to forecast monthly trends of fire incidents in San Diego

```
f_cast <- forecast(fit_ar, h=12)

autoplot(f_cast)
```




```
# forecasting to look at most recent data
autoplot(f_cast, include = 48)
```




```
head(print(summary(f_cast)))

##
## Forecast method: ARIMA(4,0,0) with zero mean
##
## Model Information:
## Series: diff_burns
## ARIMA(4,0,0) with zero mean
##
## Coefficients:
##           ar1      ar2      ar3      ar4
##       -0.9945  -0.8816  -0.5776  -0.4603
## s.e.   0.1009   0.1372   0.1354   0.1035
##
## sigma^2 estimated as 17.64:  log likelihood=-218.68
## AIC=447.37   AICc=448.21   BIC=459.09
##
## Error measures:
##                        ME     RMSE      MAE MPE MAPE      MASE        ACF1
## Training set -0.1268859 4.089907 2.786639 NaN  Inf 0.3383141 -0.07872935
##
## Forecasts:
##    Point Forecast       Lo 80      Hi 80      Lo 95     Hi 95
```

```
## 78    -5.06772694 -10.4508388  0.3153849 -13.300486  3.165032
## 79     7.26692522  -0.3251188 14.8589693  -4.344104 18.877955
## 80    -7.35322034 -14.9672621  0.2608214 -18.997892  4.291452
## 81     5.67460117  -2.0095082 13.3587106  -6.077230 17.426432
## 82    -1.02554519  -8.7651319  6.7140415 -12.862222 10.811131
## 83    -3.08055512 -11.1098465  4.9487363 -15.360297  9.199186
## 84     4.07472804  -4.2437698 12.3932259  -8.647317 16.796773
## 85    -3.35611585 -11.6813903  4.9691586 -16.088524  9.376293
## 86     1.99676959  -6.3644246 10.3579638 -10.790573 14.784113
## 87     0.03735123  -8.3614488  8.4361512 -12.807505 12.882207
## 88    -1.73458611 -10.1829581  6.7137859 -14.655256 11.186084
## 89     2.08358759  -6.4035933 10.5707685 -10.896435 15.063611

##    Point Forecast       Lo 80      Hi 80      Lo 95     Hi 95
## 78      -5.067727 -10.4508388  0.3153849 -13.300486  3.165032
## 79       7.266925  -0.3251188 14.8589693  -4.344104 18.877955
## 80      -7.353220 -14.9672621  0.2608214 -18.997892  4.291452
## 81       5.674601  -2.0095082 13.3587106  -6.077230 17.426432
## 82      -1.025545  -8.7651319  6.7140415 -12.862222 10.811131
## 83      -3.080555 -11.1098465  4.9487363 -15.360297  9.199186
```