Time Series Analysis of Atlanta Crime

James M. Wiggins
Georgia Institute of Technology
Stewart School of Industrial and Systems Engineering
ISyE 6402: Time Series Analysis Spring 2019
jameswiggins@gatech.edu

Abstract

This paper describes the final project for ISyE 6402 taught by Dr. Xiaoming Huo in the Spring of 2019. It is a time series analysis of crime data retrieved from the Atlanta Police Department (APD) website. Trend, seasonality, and stationary time series modeling techniques are all employed. Accompanying this report are presentation slides and a Jupyter Notebook.

1 Summary

The goal of this project is to model Atlanta's daily crime volume from 2009 to 2018 in order to quantitatively describe historical trends/patterns and accurately forecast Atlanta's daily crime volume one week in advance. Report-level crime data was retrieved from APD's website and aggregated by day. The last seven days of 2018 were used as testing data and the rest was used to train the models. A Splines Regression trend estimation was fit to the training data taking into account monthly and day-of-week seasonality. An Auto-Regressive model of order 14 was then fit to the training data in order to capture the remaining patterns in the stationary component of the data. These models combined resulted in prediction MAE of 10.38 and MAPE of 21.2% in forecasting the last week of 2018. These great results occured despite the test week containing two major holidays, Christmas and New Years Eve. Overall, crime in Atlanta shows a long term decreasing trend, crime is generally higher on the weekends, and Christmas/New Years Eve have on average 22% less crime than other days.

2 Analysis

After downloading the report-level crime data and aggregating it by day, the data was then separated into training and testing data sets. The original data had 3652 observations, one for each day between January 1, 2009 and December 31, 2018. The last seven days of the data were reserved as testing data leaving 3645 observations for training.

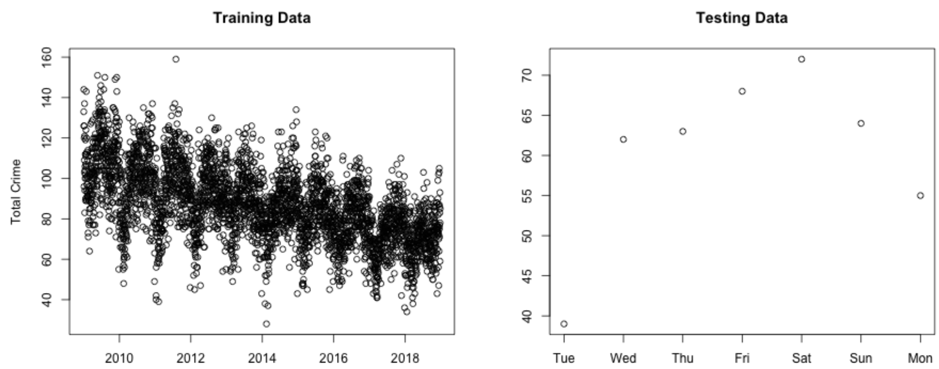


Figure 1: Training and testing data

Since the number of crimes committed in Atlanta over a period of time is inherently Poisson distributed, the data needs to be transformed in order to take a normal distribution and therefore not violate the assumptions required for further modeling. The classic transformation of $\sqrt{x + 3/8}$ was employed. This transformation will be reversed later when making predictions.

2.1 Trend

Numerous models were explored in order to best estimate the trend present in the training data including Splines Regression, Locally Weight Polynomial, Parametric Regression, and Moving Average. Splines Regression was selected as the final model for its smooth variation over time that accurately reflects the downward pattern. An R-squared value of 0.331 for this model shows that this trend alone accounts for 33.1% of the variation in daily crime volume.

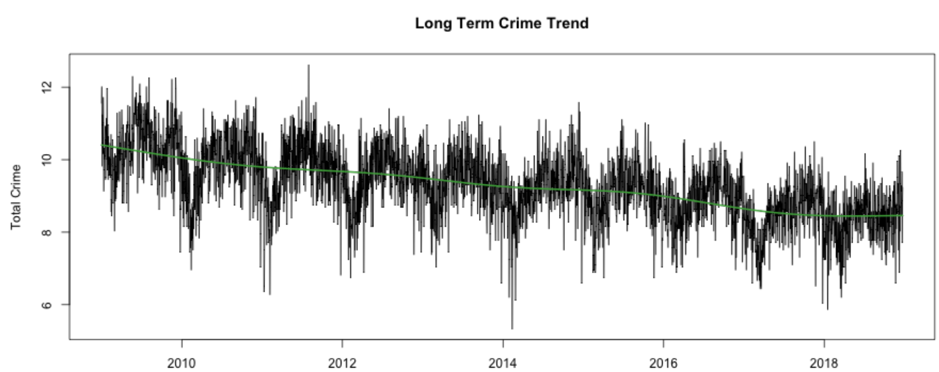


Figure 2: Estimating the trend in Atlanta's daily crime

2.2 Seasonality

After accounting for the downward trend, month and day-of-week seasonality were added to the splines regression model. Both of these factors were highly significant as shown by the p-values of their model parameters (see Appendix). An R-squared value of 0.889 shows that accounting for trend and seasonality accounts for approximately 90% of the variation in the volume of daily crime in the city of Atlanta, Georgia.

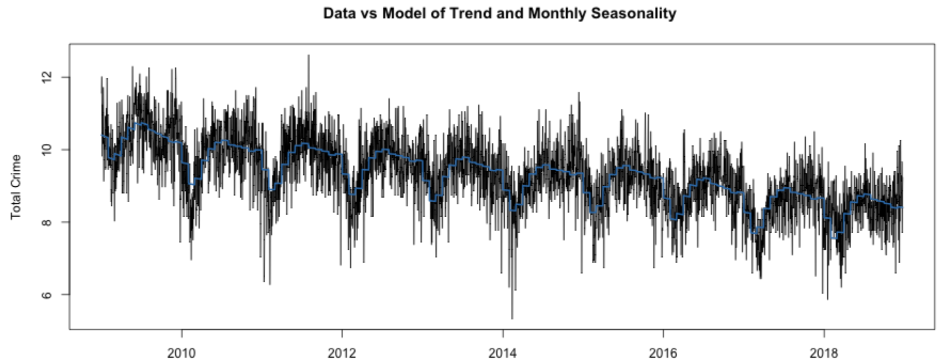


Figure 3: Modeling the trend and monthly seasonality in ATL crime data

2.3 Time Series Model

The final stage of this modeling process is modeling the residual stationary time series process. An Auto-Regressive model of order 14 was found to minimize the Akaike information criterion (AIC). More complex ARMA models were fit to the data but did not increase performance so were rejected in favor of the less complex model. The Auto-Regressive model was tested for uncorrelated residuals with both Box-Pierce and Ljung-Box tests and were found to not result in correlated residual values.

3 Conclusion

Combining the models for trend and seasonality with the AR(14) model results in prediction MAE 10.38 and MAPE 21.2% when compared to the testing data. The first and last days for the testing data are both holidays (Christmas Day and New Years Eve respectively) and exhibited significantly lower crime than the forecasting model showed. In the data, Christmas and New Years Eve typically have 22% less crime than other days, so it makes sense that the model would overestimate the observed volume for those days.

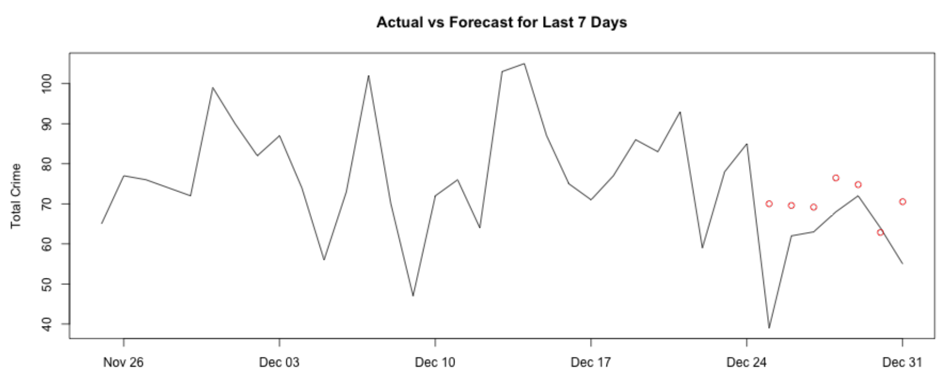


Figure 4: Final model forecast for testing data vs data

4 References

[1] Brockwell, P. J., & Davis, R. A. (2016). Introduction to time series and forecasting. Switzerland: Springer.

[2] Shumway, R. H., & Stoffer, D. S. (2011). Time series analysis and its applications: With R examples (3rd ed.). New York: Springer.

Appendix

April 28, 2019

```
In [1]: # Load necessary packages
        library(plyr)
        library(dplyr)
        library(repr)
        library(mgcv)
        library(lubridate)
```

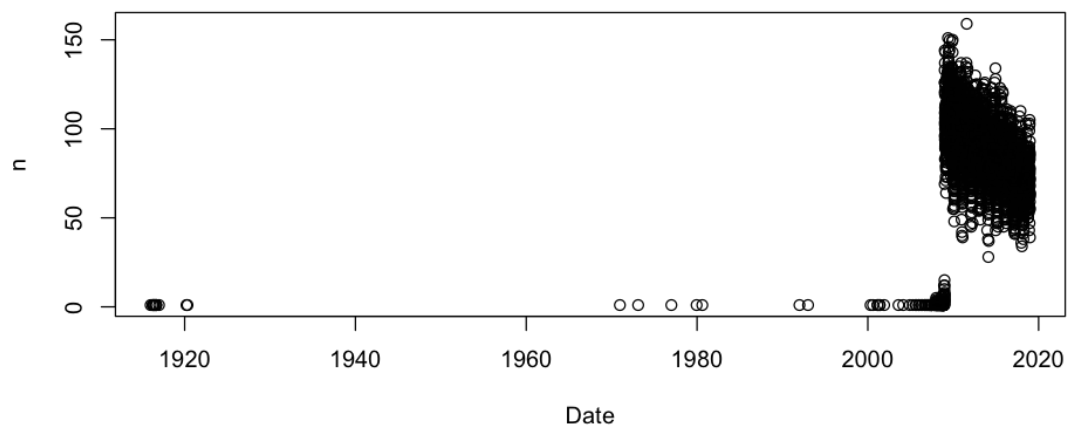
```
Attaching package: dplyr

The following objects are masked from package:plyr:

    arrange, count, desc, failwith, id, mutate, rename, summarise,
    summarize

The following objects are masked from package:stats:

    filter, lag

The following objects are masked from package:base:

    intersect, setdiff, setequal, union

Loading required package: nlme

Attaching package: nlme

The following object is masked from package:dplyr:

    collapse

This is mgcv 1.8-26. For overview type 'help("mgcv-package")'.

Attaching package: lubridate

The following object is masked from package:plyr:
```

```
    here

The following object is masked from package:base:

    date
```

```
In [2]: CrimeData <- read.csv("COBRA-2009-2018.csv", header=T)
        CrimeData <- select(CrimeData, c(Report.Number, Occur.Date))
        names(CrimeData) <- c("Report.Number", "Date")
        tail(CrimeData)
```

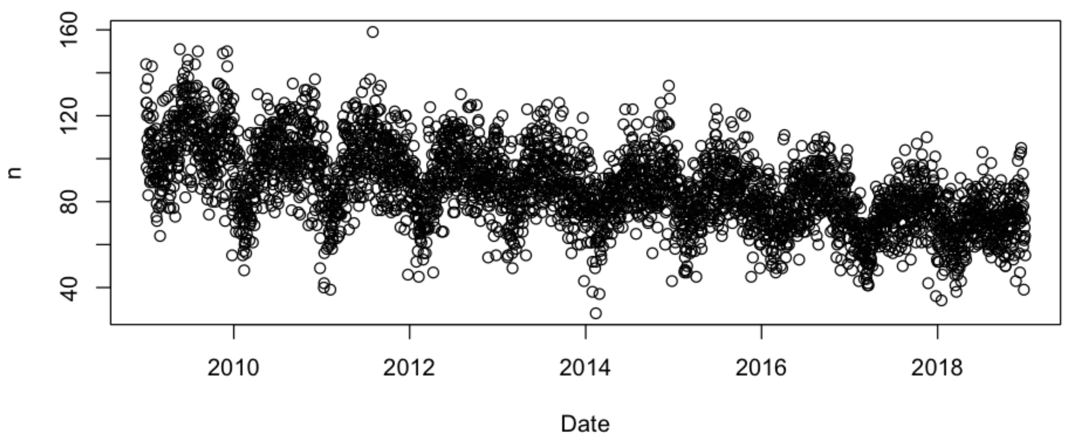
	Report.Number	Date
317899	183651676	2018-12-31
317900	183651752	2018-12-31
317901	183652121	2018-12-31
317902	183652271	2018-12-31
317903	183650457	2018-12-31
317904	183651734	2018-12-26

```
In [3]: # View Raw Count Data
        tsData <- count(CrimeData, Date)
        tsData$Date <- as.Date(tsData$Date)
        options(repr.plot.width=8, repr.plot.height=4)
        plot(tsData)
```




```
In [4]: # Remove time points before 2009
        tsData <- filter(tsData, Date >= as.Date("2009-01-01"))
        plot(tsData)
        head(tsData)
```

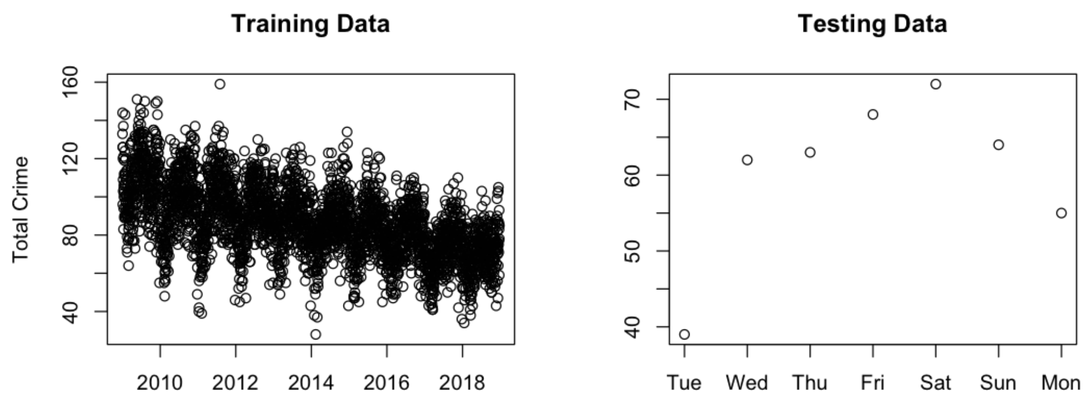
Date	n
2009-01-01	133
2009-01-02	144
2009-01-03	126
2009-01-04	96
2009-01-05	126
2009-01-06	120



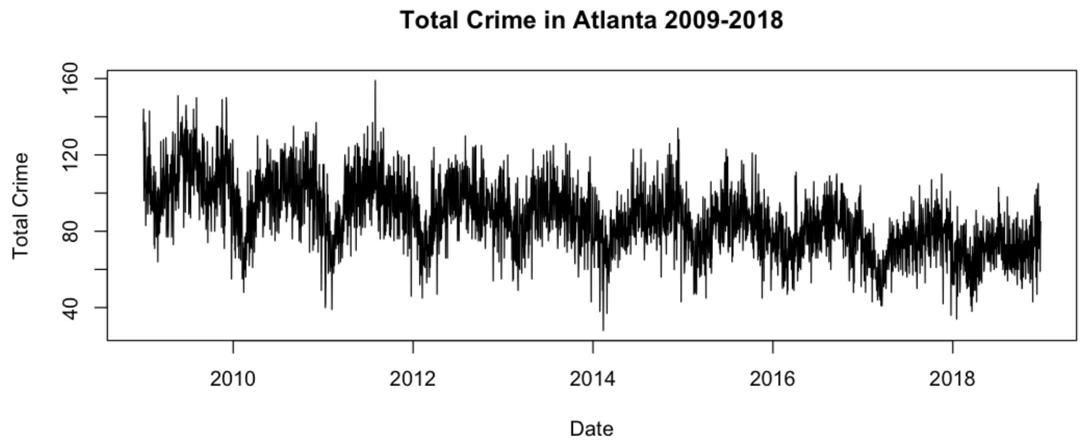
0.0.1 The goal of this project will be to forecast the crime one a per day one week in advance. So I will withhold the last weeks worth of data as the testing data and train on the rest.

```
In [5]: options(repr.plot.width=9, repr.plot.height=4)
        par(mfrow=c(1,2))
        training <- tsData[1:(length(tsData$Date)-7),1:2]
        testing <- tsData[(length(tsData$Date)-6):length(tsData$Date),1:2]
        plot(training, main='Training Data', ylab="Total Crime",xlab="")
        plot(testing, main='Testing Data', ylab="",xlab="")
        n = length(tsData$Date)
        nfit = length(training$Date)
        ntest = length(testing$Date)

        dev.copy(png,'training.png',width=900, height=400)
        dev.off()
```

quartz_off_screen: 3
pdf: 2

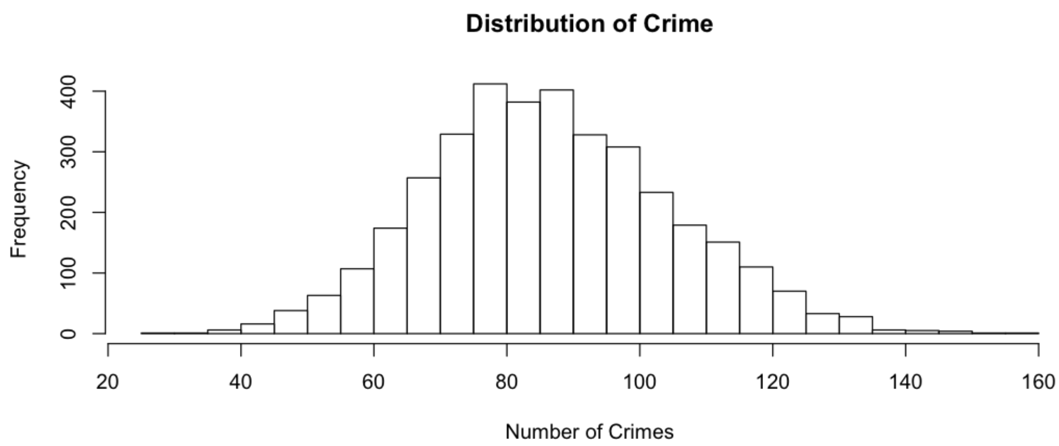



```
In [6]: # Convert to time series data
        Crime = as.vector(training$n)
        Crime <- ts(Crime, start=2009, frequency=365.25)
        ts.plot(Crime, main="Total Crime in Atlanta 2009-2018", xlab="Date", ylab="Total Crime
```



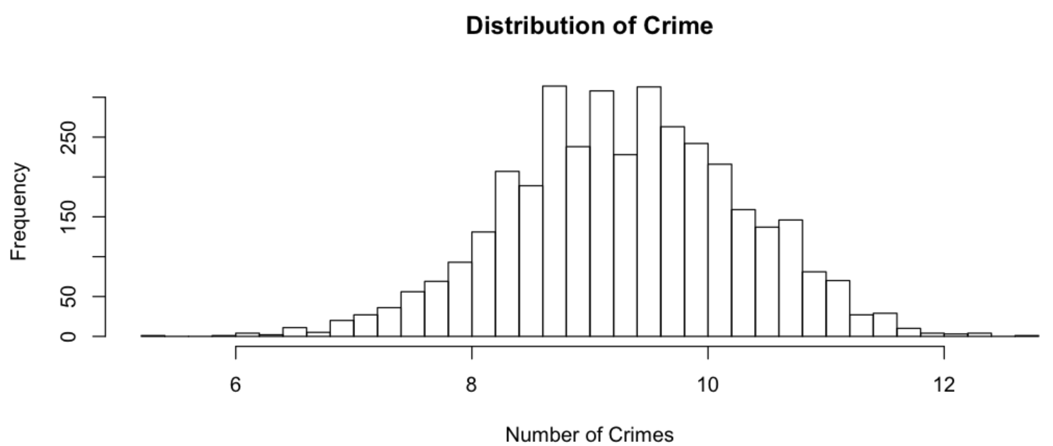
0.0.2 From this plot we can detect a downward trend which is good, we want crime to decrease generally speaking. I will remove this trend as part of the TS modeling process. We also see a cyclical pattern in the plot which I suspect has to do with yearly seasonality. This seasonality will be estimated and removed shortly.

```
In [7]: hist(Crime, nclass=20, main='Distribution of Crime', xlab = 'Number of Crimes')
```



0.0.3 Next, I will estimate the trend using Local Polynomail Trend estimation and Splines trend estimation to see which fits the data better.

In [8]:
```
Crimes = sqrt(Crime + 3/8)
hist(Crimes, nclass=30, main='Distribution of Crime', xlab = 'Number of Crimes')
```



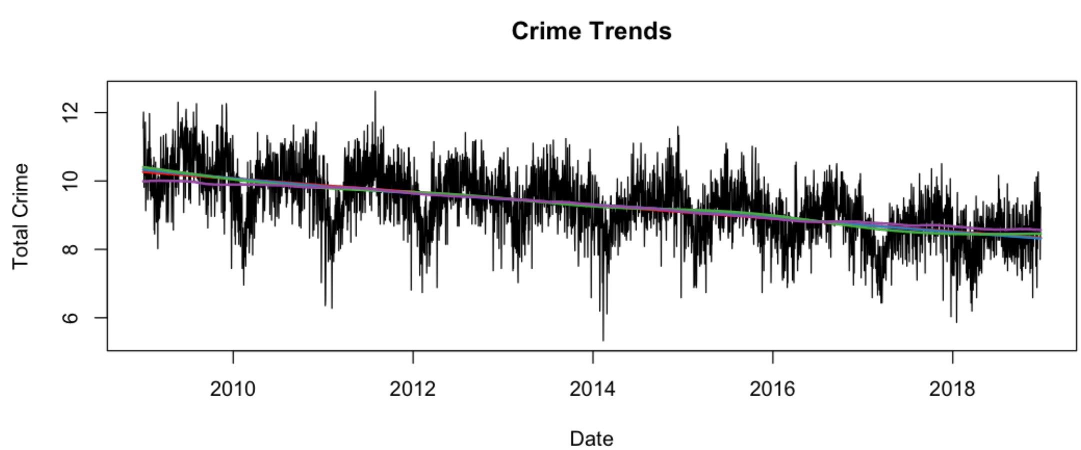
0.0.4 Since count data is inherently poisson distributed, I will apply a classic transformation of the count data to ensure it is normally distributed. This transformation will reduce the variability in the time series so it will not violate the non-constant variance assumption required for linear modeling.

In [9]:
```
index = c(1:n)
index = index/max(index)
```

```
time.pts = index[1:nfit]
# Loess
loc.fit = loess(Crimes~time.pts)
vol.fit.loc = ts(fitted(loc.fit), start=2009, frequency=365.25)
# Splines
gam.fit = gam(Crimes~s(time.pts))
vol.fit.gam = ts(fitted(gam.fit), start=2009, frequency=365.25)
# Moving Average
mav.fit = ksmooth(time.pts, Crimes, kernel = 'box')
vol.fit.mav = ts(mav.fit$y, start=2009, frequency=365.25)
```

In [10]:
```
# Parametric Regression
x1 = time.pts
x2 = time.pts^2
lm.fit = lm(Crimes~x1+x2)
vol.fit.lm = ts(fitted(lm.fit), start=2009, frequency=365.25)
```

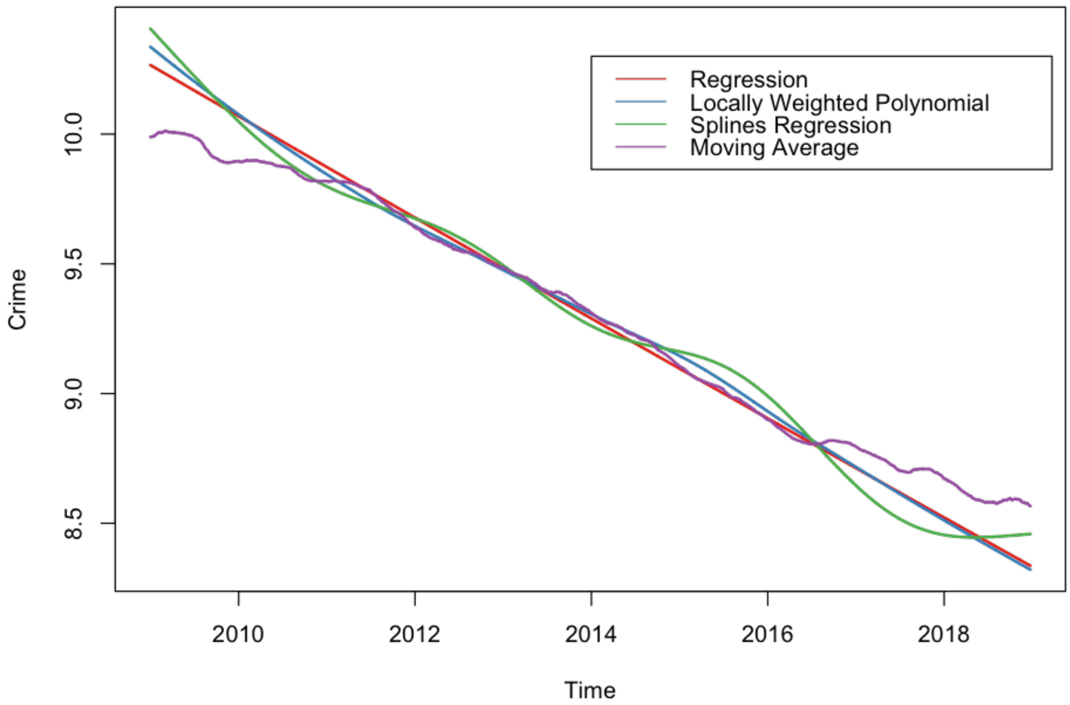
In [11]:
```
# plot results
ts.plot(Crimes, main="Crime Trends", xlab="Date", ylab="Total Crime", type='l')
lines(vol.fit.lm, col='#e41a1c', lwd=2)
lines(vol.fit.loc, col='#377eb8', lwd=2)
lines(vol.fit.gam, col='#4daf4a', lwd=2)
lines(vol.fit.mav, col='#984ea3', lwd=2)
```



0.0.5 All trends are very similar showing generally decreasing pattern and smooth variations over time. It is a little hard to differential between them so lets plot the trends without the time series.

In [12]:
```
# Compare all estimated trends
options(repr.plot.width=8, repr.plot.height=6)
all.val = c(vol.fit.lm,vol.fit.loc,vol.fit.gam,vol.fit.mav)
```

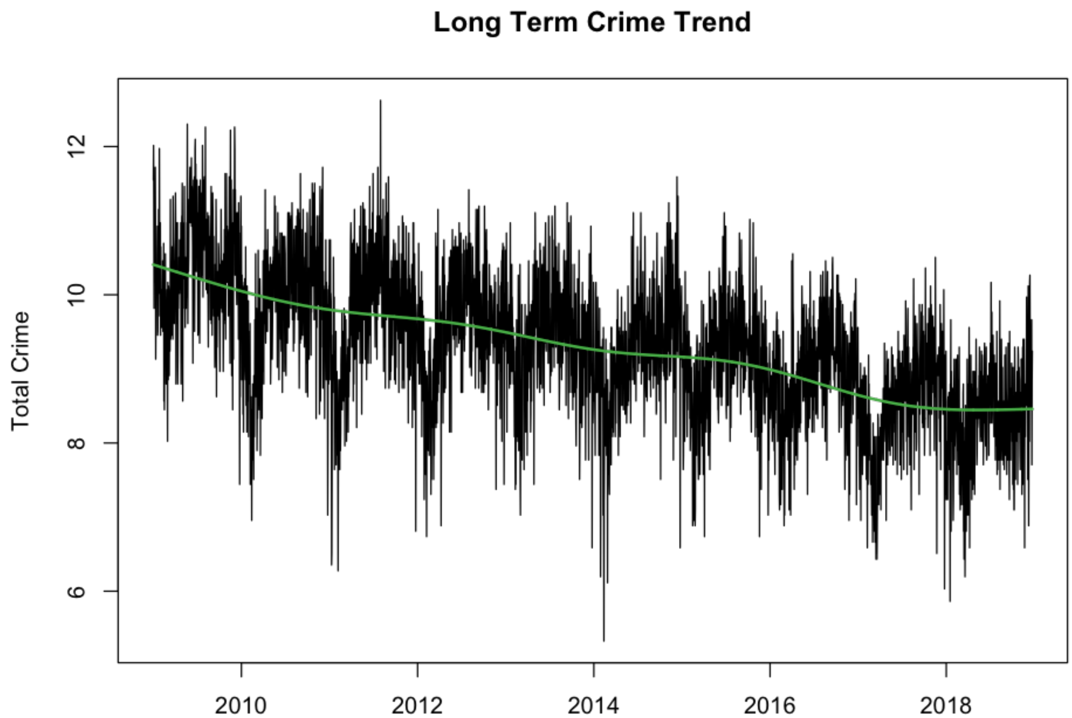
```
ylim = c(min(all.val),max(all.val))
ts.plot(vol.fit.lm,lwd=2,col='#e41a1c',ylim=ylim,ylab="Crime")
lines(vol.fit.loc, col='#377eb8', lwd=2)
lines(vol.fit.gam, col='#4daf4a', lwd=2)
lines(vol.fit.mav, col='#984ea3', lwd=2)
legend(x=2014,y=10.3,legend=c('Regression', 'Locally Weighted Polynomial','Splines Reg
col=c('#e41a1c','#377eb8','#4daf4a','#984ea3'))
```



0.0.6 Splines Regression looks good to me! Let's see if the trend is significant...

```
In [13]: # plot results
         ts.plot(Crimes, main="Long Term Crime Trend",xlab = "", ylab="Total Crime", type='l')
         lines(vol.fit.gam, col='#4daf4a', lwd=2)
         dev.copy(png,'splines.png',width=900, height=400)
         dev.off()
```

quartz_off_screen: 3
pdf: 2




```
In [14]: summary(gam.fit)


Family: gaussian
Link function: identity

Formula:
Crimes ~ s(time.pts)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.29479    0.01345   691.3   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Approximate significance of smooth terms:
              edf Ref.df     F p-value
s(time.pts) 7.046  8.109 214.7  <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

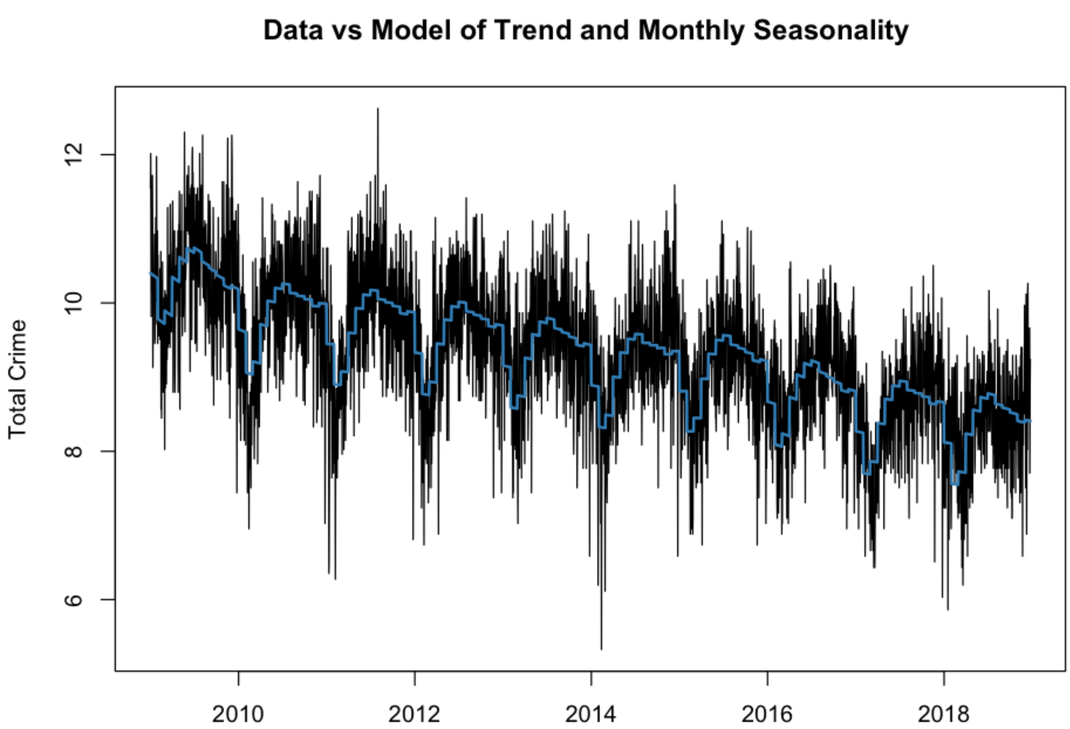
```
R-sq.(adj) =  0.323   Deviance explained = 32.4%
GCV = 0.66042  Scale est. = 0.65896   n = 3645
```

0.0.7 This output shows us that the smooth trend is highly statistically significant because p-value is very small. Additionally, the R-squared value is 33.1% indicating the approximately one third of the variation in the data is due to trend alone

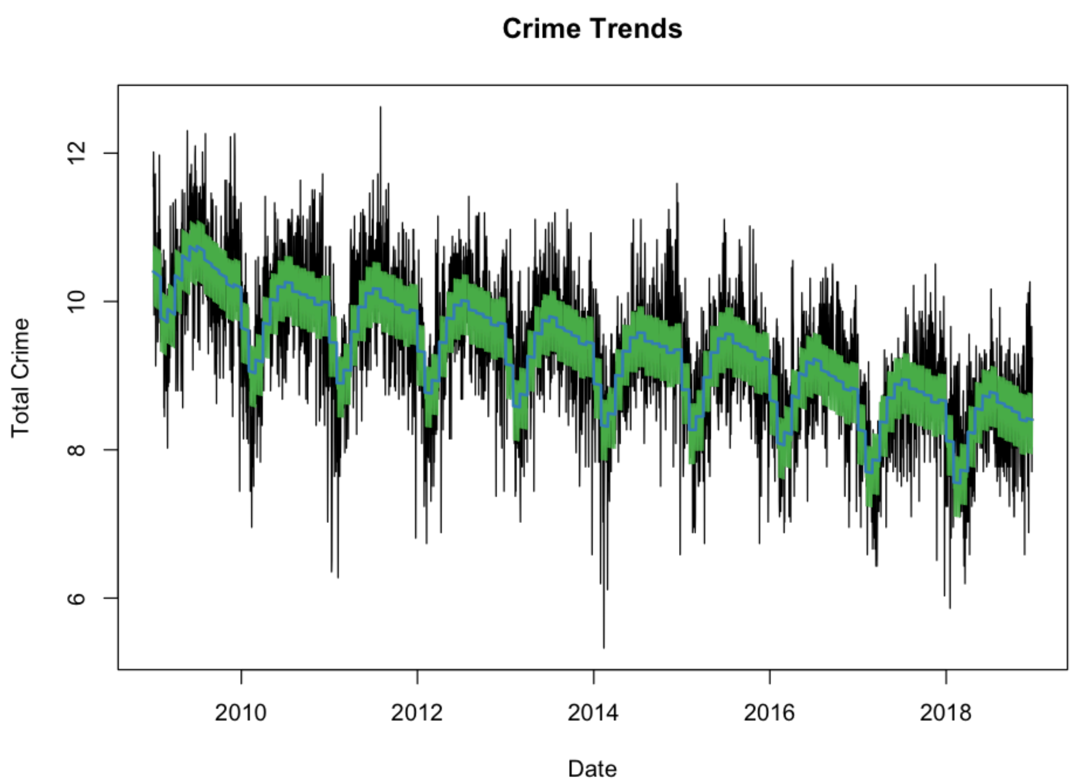
0.0.8 Next, let's add monthly seasonality

```
In [15]: month = as.factor(format(training$Date,"%b"))
         MonthlyS = gam(Crimes~s(time.pts) + month)
         FittedMonthly = fitted(MonthlyS)
         # plot results
         options(repr.plot.width=8, repr.plot.height=6)
         plot(training$Date,Crimes,type='l', main="Data vs Model of Trend and Monthly Seasonali
         lines(training$Date,FittedMonthly,col='#377eb8',lwd=2)

         dev.copy(png,'seasonality.png',width=900, height=400)
         dev.off()
```

quartz_off_screen: 3
pdf: 2



```
In [16]: ## Next, Add day-of-the-week seasonality
         options(repr.plot.width=8, repr.plot.height=6)

         month = as.factor(format(training$Date,"%b"))
         week = as.factor(weekdays(training$Date))
         WnMSeasonality = gam(Crimes~ s(time.pts) + month + week)


         FittedWnM = fitted(WnMSeasonality)
         ## Compare the two fits: with & without day-of-the-week seasonality
         plot(training$Date, Crimes, type='l', main="Crime Trends", xlab="Date", ylab="Total C
         lines(training$Date,FittedWnM,col='#4daf4a',lwd=1)
         lines(training$Date,FittedMonthly,col='#377eb8',lwd=2)
```




```
In [17]: ## Does the addition of seasonality of day of the week add predictive power?
         lm.fit.seastr.1 = lm(Crimes~month)
         lm.fit.seastr.2 = lm(Crimes~month+week)
         anova(lm.fit.seastr.1,lm.fit.seastr.2)
         NoTrend = fitted(lm.fit.seastr.2)
```

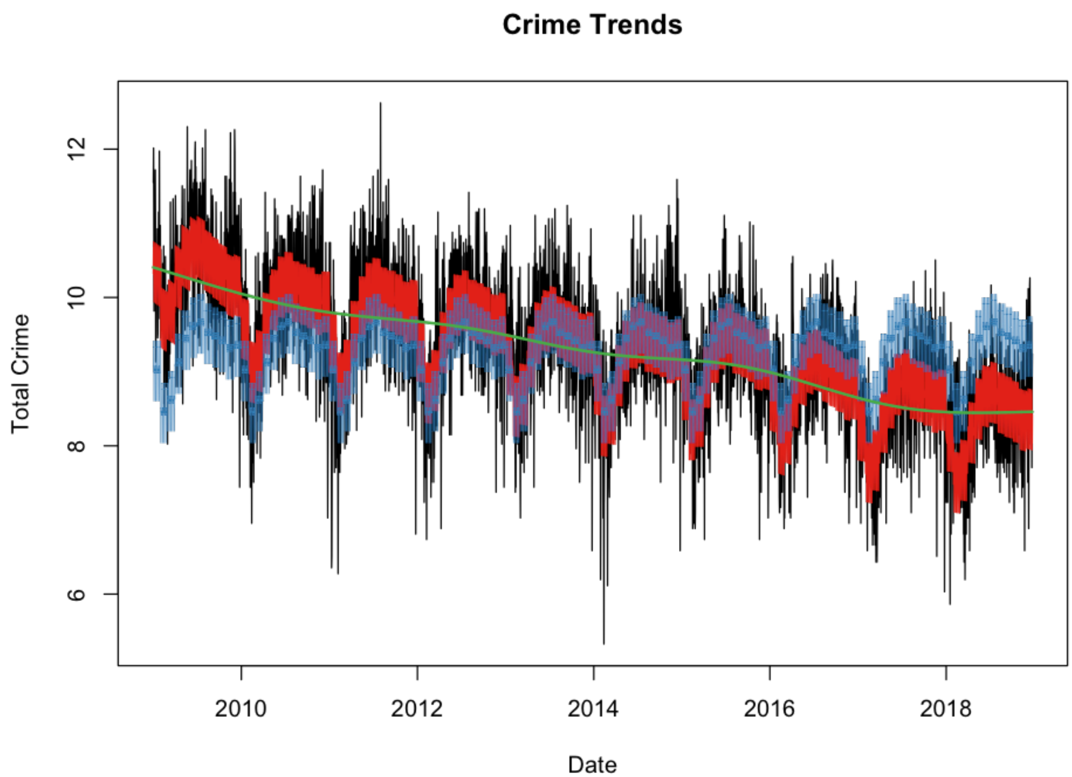
Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
3633	3069.977	NA	NA	NA	NA
3627	2856.031	6	213.9463	45.28331	1.037252e-53

0.0.9 The p-value is small so we can conclude both weekly and monthly seasonality are helpful in predicting volume of crime in Atlanta.

```
In [18]: summary(lm.fit.seastr.2)


Call:
lm(formula = Crimes ~ month + week)

Residuals:
    Min      1Q  Median      3Q     Max
-3.3745 -0.5973  0.0156  0.5964  3.0466

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    9.51039    0.06260 151.919  < 2e-16 ***
monthAug       0.40436    0.07187   5.626 1.98e-08 ***
monthDec       0.23227    0.07227   3.214  0.00132 **
monthFeb      -0.66412    0.07360  -9.023  < 2e-16 ***
monthJan      -0.09840    0.07187  -1.369  0.17102
monthJul       0.53911    0.07187   7.501 7.90e-14 ***
monthJun       0.48650    0.07245   6.715 2.18e-11 ***
monthMar      -0.50599    0.07187  -7.041 2.28e-12 ***
monthMay       0.31858    0.07187   4.433 9.57e-06 ***
monthNov       0.19077    0.07245   2.633  0.00850 **
monthOct       0.29550    0.07187   4.112 4.01e-05 ***
monthSep       0.35427    0.07245   4.890 1.05e-06 ***
weekMonday    -0.33700    0.05498  -6.129 9.76e-10 ***
weekSaturday  -0.08850    0.05498  -1.610  0.10754
weekSunday    -0.80748    0.05498 -14.687  < 2e-16 ***
weekThursday  -0.43108    0.05498  -7.841 5.85e-15 ***
weekTuesday   -0.37820    0.05501  -6.876 7.24e-12 ***
weekWednesday -0.40432    0.05501  -7.350 2.43e-13 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.8874 on 3627 degrees of freedom
Multiple R-squared:  0.1948,Adjusted R-squared:  0.191
F-statistic: 51.61 on 17 and 3627 DF,  p-value: < 2.2e-16
```

0.0.10 This output shows that all monthly factors except for January are statistically significant, and all day-of-week factors are significant except for Saturday. The R-squared value of this model is 88.9 which means that almost 90% of the variability in our data can be explained by Month and Day-of-Week alone.

In [19]:
```
## Compare with & without trend
options(repr.plot.width=8, repr.plot.height=6)
plot(training$Date,Crimes,type='l', main="Crime Trends", xlab="Date", ylab="Total Cri
lines(training$Date,FittedWnM,lwd=1,col="#e41a1c")
lines(training$Date,NoTrend,lwd=.5,col="#377eb8")
lines(training$Date,vol.fit.gam,lwd=2,col="#4daf4a")
```



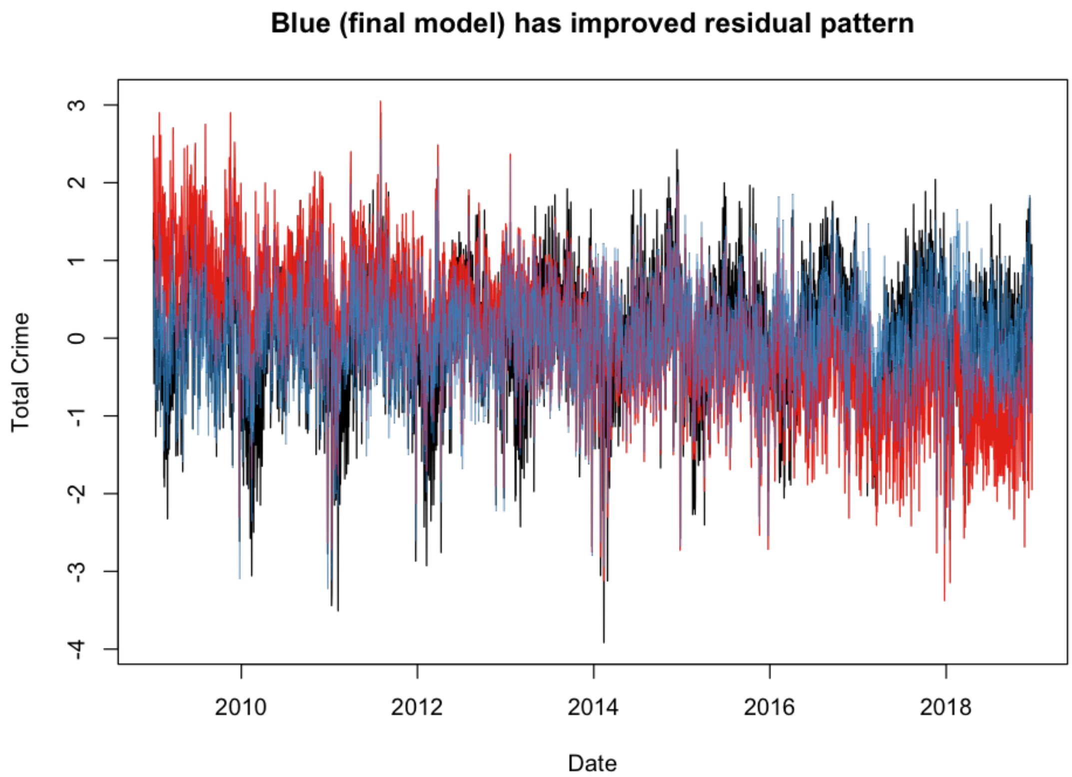
0.0.11 Black = data, Red = full model, Blue = seasonality but no trend, Green = trend

0.0.12 Next, let's look at stationarity

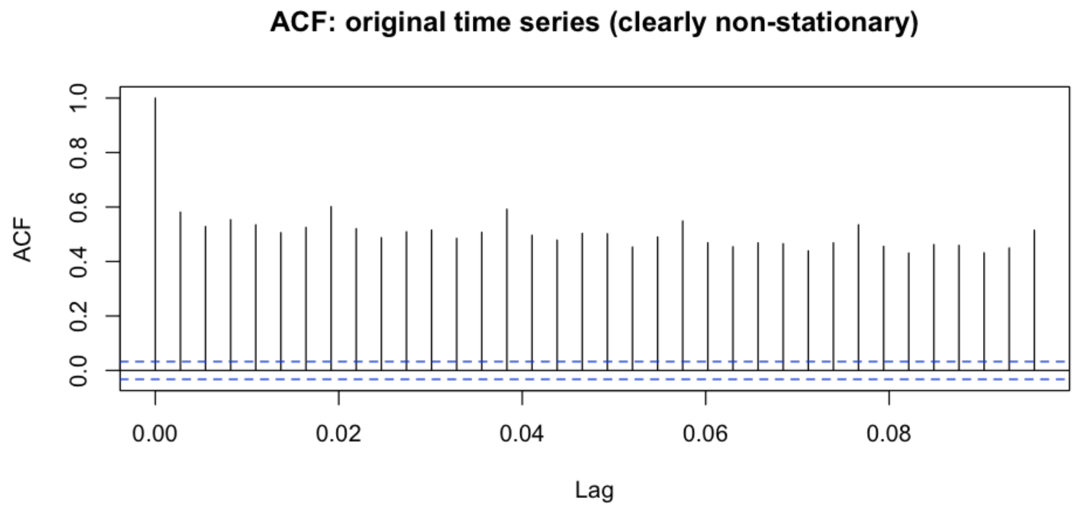
In [20]:
```
## Residual Process: Trend Removal
resid.1 = Crimes-vol.fit.gam
## Residual Process: Stationarity Removal
resid.2 = Crimes-NoTrend
## Residual Process: Trend & Stationarity Removal
residuals = Crimes-FittedWnM
```

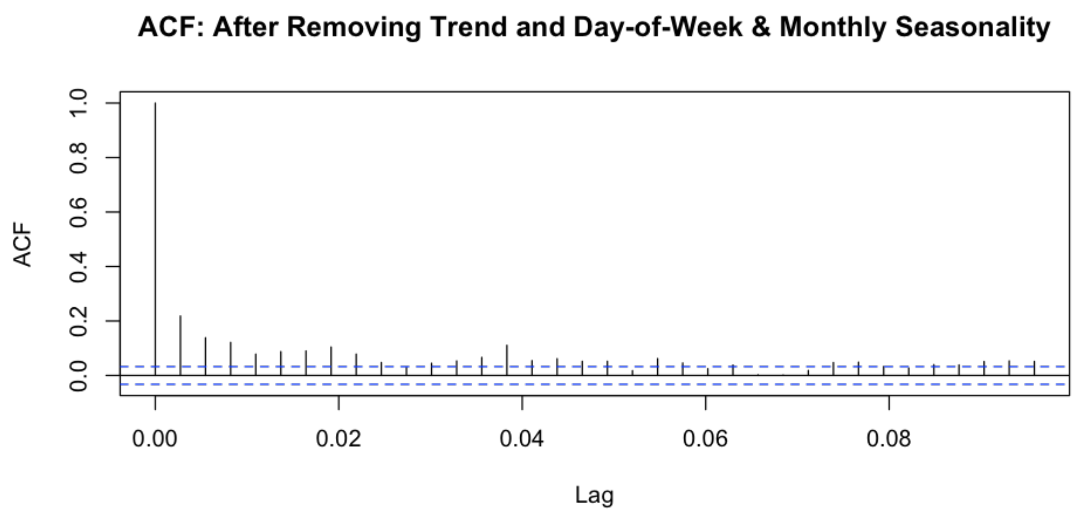
```
        y.min = min(c(resid.1,resid.2,residuals))
        y.max = max(c(resid.1,resid.2,residuals))
```

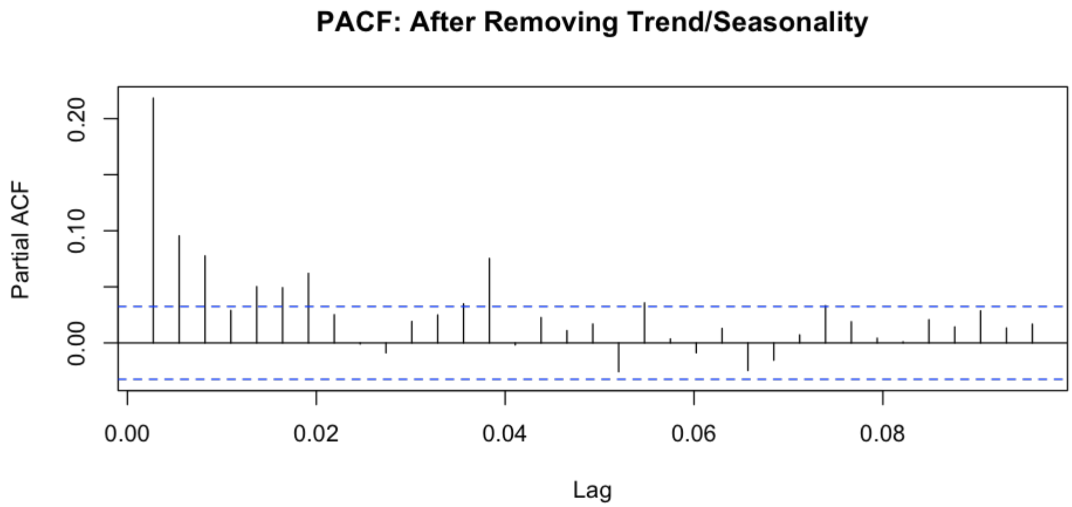
```
In [21]: #Graph the three Models
         options(repr.plot.width=8, repr.plot.height=6)
         plot(training$Date,resid.1,type='l', main="Blue (final model) has improved residual pa
         lines(training$Date,resid.2,lwd=1,col="#e41a1c")
         lines(training$Date,residuals,lwd=.5,col="#377eb8")
```




```
In [22]: options(repr.plot.width=8, repr.plot.height=4)
         acf(Crimes, main='ACF: original time series (clearly non-stationary)')
         acf(residuals, main="ACF: After Removing Trend and Day-of-Week & Monthly Seasonality")
         pacf(residuals, main='PACF: After Removing Trend/Seasonality')
```


ACF: original time series (clearly non-stationary)
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.02
0.04
0.06
0.08
Lag

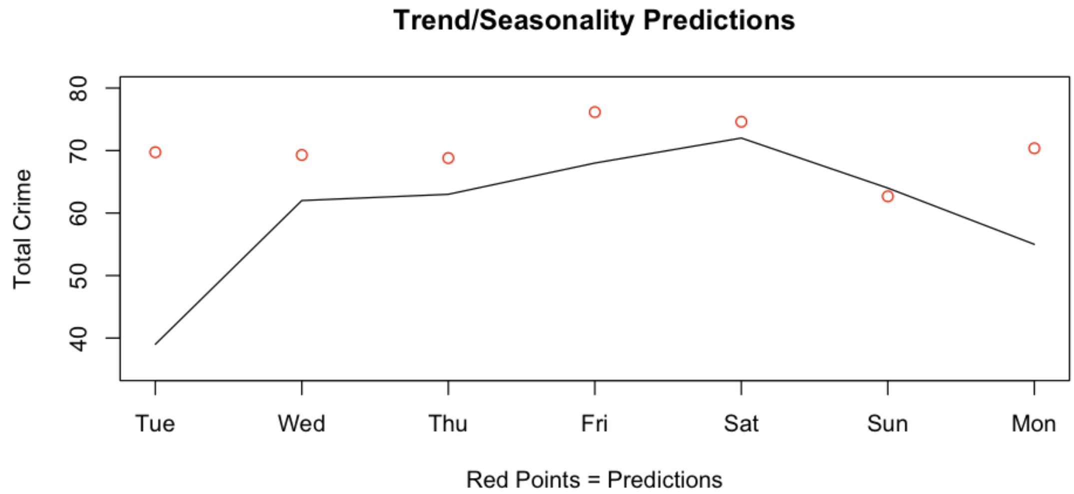

ACF: After Removing Trend and Day-of-Week & Monthly Seasonality
ACF
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.02
0.04
0.06
0.08
Lag



0.0.13 This acf plot shows the acf values of the residual time series after removing trend and seasonality. Both the ACF and PACF decrease rapidly for increasing lags which indicates that both the trend and seasonality have been removed.

0.0.14 Let's go ahead and predict based off of trend and seasonality alone

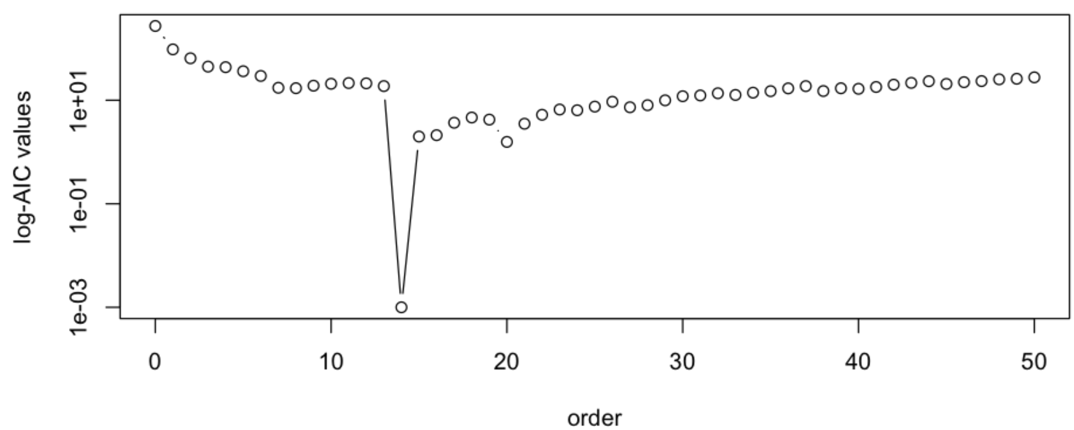
In [23]:
```
options(repr.plot.width=8, repr.plot.height=4)

newdata=data.frame(time.pts=index[(nfit+1):n],
                   week = as.factor(weekdays(testing$Date)),
                   month = as.factor(format(testing$Date,"%b")))

predictions = predict(WnMSeasonality, newdata, interval=c("prediction"))^2-3/8
#predictions$Date <- testing$Date
preds = data.frame(Date = testing$Date, n = predictions)
plot(testing, type = 'l',ylim=c(35,80), ylab="Total Crime", main='Trend/Seasonality P
points(preds, col='red')
```



0.0.15 Findings so far, both day-of-week and monthly seasonality are statistically significant. After removing these components the time series becomes stationary.

0.0.16 Next, fit an AR Model

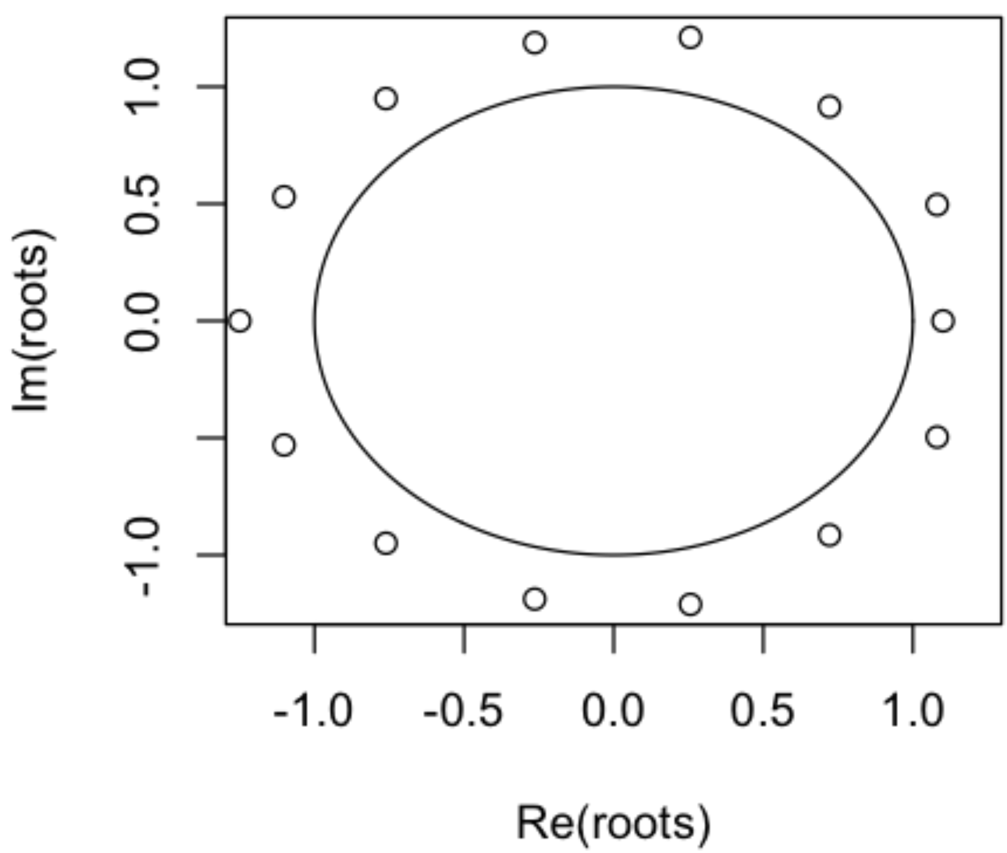
```
In [24]: mod = ar(residuals, order.max=50)
         plot(c(0:50),mod$aic+0.001,type="b",log="y", xlab='order',ylab='log-AIC values')
```



0.0.17 for the AR model, an order of 14 is the optimal order for minimizing AIC.

0.0.18 Next, lets see whethere this process is both stationary and causal by inspecting the roots of the polynomial

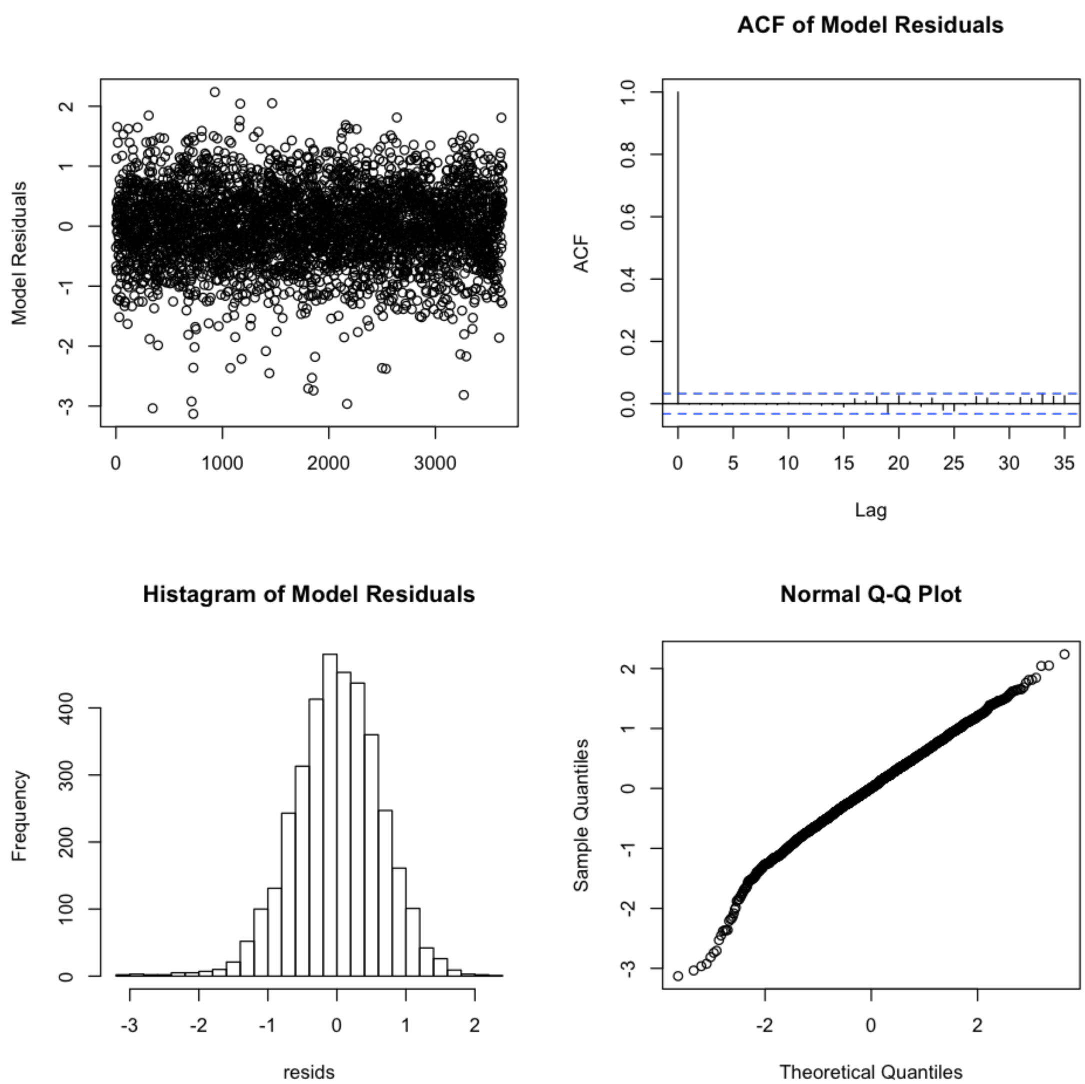
In [25]:
```
## are the roots of the fitter AR within the unit circle?
options(repr.plot.width=4, repr.plot.height=4)
## extract roots from the model output:
roots = polyroot(c(1,(-mod$ar)))
## Adjust the x & y axis limits to include the full circle
plot(roots,xlim=c(-1.2,1.2),ylim=c(-1.2,1.2))
# draw a unit cirle
lines(complex(arg=seq(0,2*pi,len=300)))
```



0.0.19 Since the roots are not on the unit circle this model is stationary, and since the roots are outside the unit circle this process is causal

0.0.20 Next let's look at the residual process

In [26]:
```
## obtain the standardized residuals
options(repr.plot.width=8, repr.plot.height=8)
resids = mod$resid[(mod$order+1):length(mod$resid)]
par(mfrow=c(2,2))
plot(resids,xlab="",ylab="Model Residuals")
acf(resids,main="ACF of Model Residuals")
hist(resids, main='Histagram of Model Residuals', nclass=30)
qqnorm(resids)
```



0.0.21 The residual plot does not show any discernable patterns and displays constant variance. The ACF plot displays a stationary process. The histogram shows approximately nornally distributed residuals. From the Q-Q plot we can confirm that the residuals are approximately nornally distributed.

0.0.22 Next, let's fit an ARMA model to see if we can get a better fit

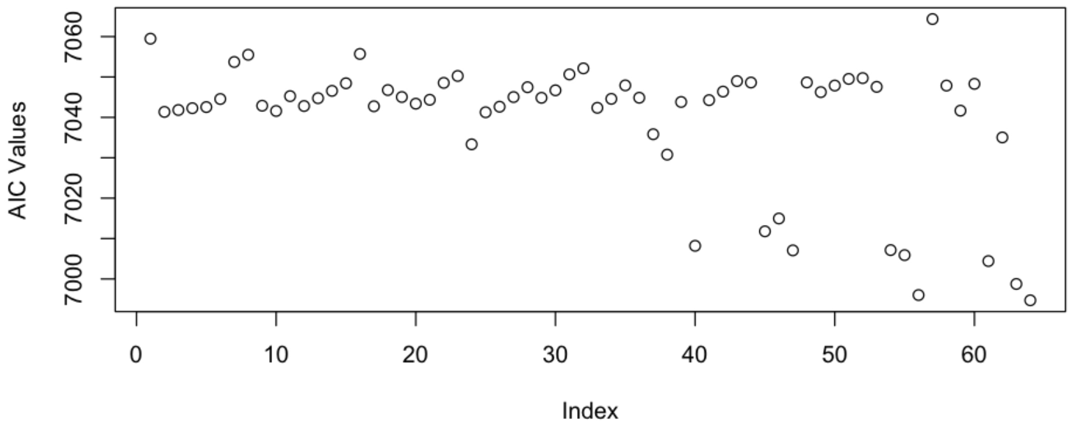
```
In [27]: #library(TSA)
```

```
In [28]: # Using minimum AIC
         n = length(residuals)
         norder = 8
         p = c(1:norder)
         q = c(1:norder)
         aic = matrix(0,length(p),length(q))
```

```
In [29]: # Start the clock!
         ptm <- proc.time()

         options(warn=-1)
         for(i in 1:length(p)){
             for(j in 1:length(q)){
                 modij = arima(residuals, order=c(p[i],0,q[j]), method='ML')
                 aic[i,j]=modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
             }
         }
         options(warn=0)

         # Stop the clock
         proc.time() - ptm
```

```
   user  system elapsed
236.633   1.432 243.786
```

```
In [30]: ## Which order to select?
         options(repr.plot.width=8, repr.plot.height=4)
         aicv = as.vector(aic)
         plot(aicv,ylab='AIC Values')
```




```
In [31]: indexp=rep(p,each=length(p))
         indexq=rep(q,length(q))
         indexaic = which(aicv==min(aicv))
         porder = indexp[indexaic]
         qorder = indexq[indexaic]
         print("best p and q:")
         porder
         qorder
         ## Final Model
         arma_model = arima(residuals,order=c(porder,0,qorder),method='ML')
```

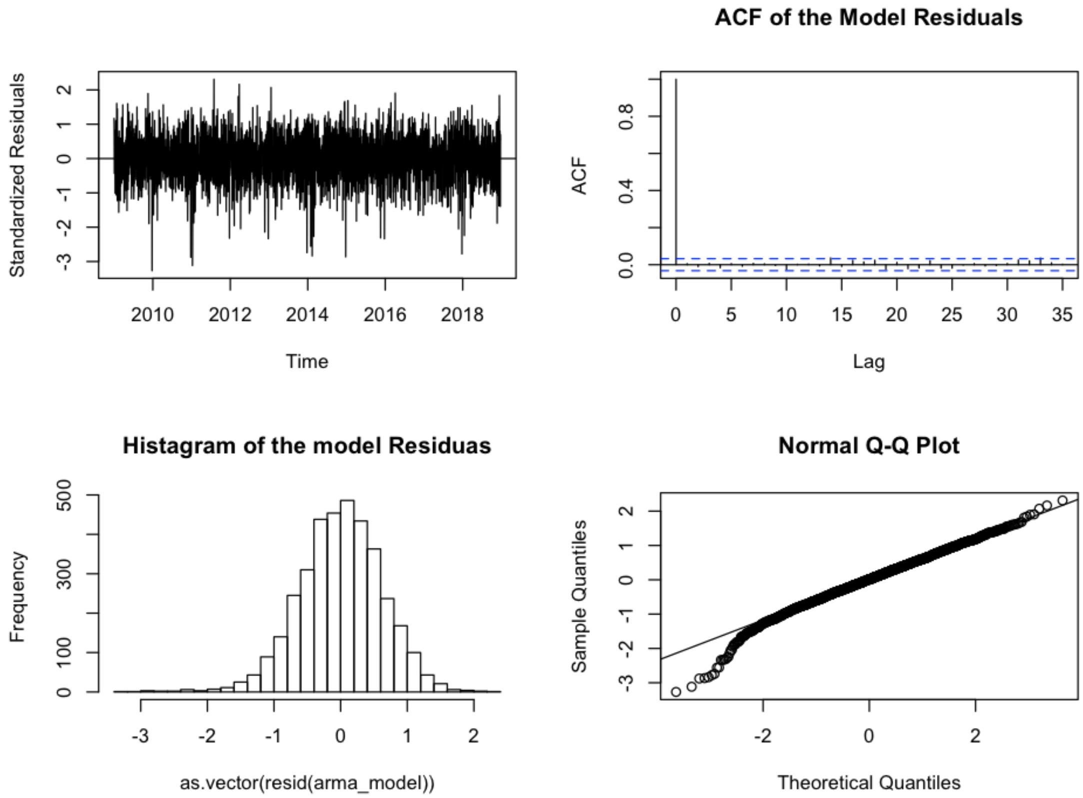
```
[1] "best p and q:"
```

8

8

```
Warning message in arima(residuals, order = c(porder, 0, qorder), method = "ML"):
possible convergence problem: optim gave code = 1
```

```
In [32]: ## Residual Analysis
         options(repr.plot.width=8, repr.plot.height=6)
         par(mfrow=c(2,2))
         plot(resid(arma_model), ylab='Standardized Residuals')
         abline(h=0)
         acf(as.vector(resid(arma_model)),main="ACF of the Model Residuals")
         hist(as.vector(resid(arma_model)),main='Histagram of the model Residuas',nclass=30)
         qqnorm(resid(arma_model))
         qqline(resid(arma_model))
```



0.0.23 since this model does not seem to perform any better, I will choose to keep the simpler AR(14) Model

0.0.24 Are the residuals uncorrelated??

0.0.25 For this test the null hypothesis is the residuals from the model fit are uncorrelated versus the alternative hypothesis that they are correlated. Thus, for such tests we seek large P values.

```
In [33]: ### Test for uncorrelated for AR(14) Model
         Box.test(mod$resid, lag = (porder+qorder+1),type = "Box-Pierce", fitdf=(porder+qorder)
         Box.test(mod$resid, lag = (porder+qorder+1),type = "Ljung-Box", fitdf=(porder+qorder)
```

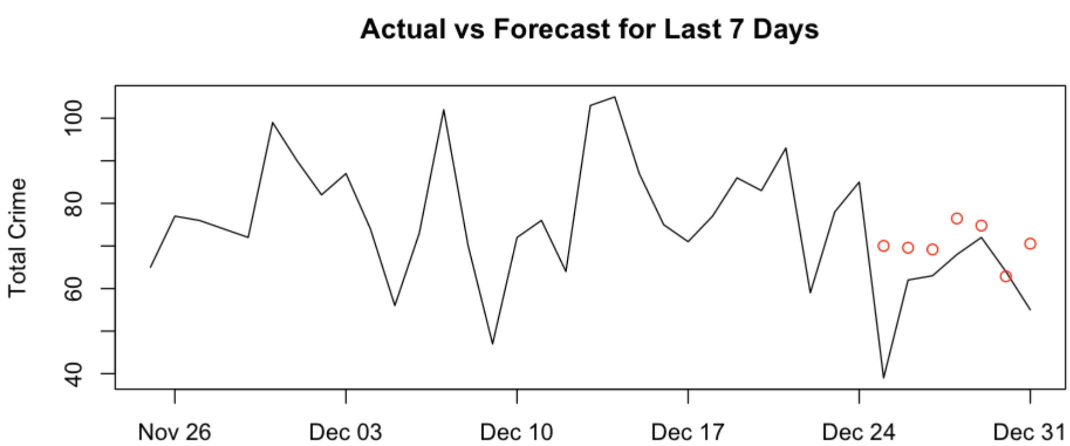
```
Box-Pierce test

data:  mod$resid
X-squared = 1.7534, df = 1, p-value = 0.1854
```

```
Box-Ljung test

data:  mod$resid
X-squared = 1.7616, df = 1, p-value = 0.1844
```

0.0.26 Time to start predicting things!!

```
In [34]: ## Predictions for AR(14) Model
         ARPredictions = predict(mod,n.ahead=7)
         final_Predictions = data.frame(Date = testing$Date, predictions = ARPredictions$pred
         ubound = ARPredictions$pred+1.96*ARPredictions$se
         lbound = (ARPredictions$pred-1.96*ARPredictions$se)

         options(repr.plot.width=8, repr.plot.height=4)
         plot(filter(tsData, Date >= as.Date("2018-11-25")), type='l', main="Actual vs Forecas
         points(final_Predictions, col='red')
         dev.copy(png,'final.png',width=900, height=400)
         dev.off()
```

quartz_off_screen: 3
pdf: 2




```
In [35]: ### Calculate Mean Absolute Error
         predicted = final_Predictions$predictions
         obs = filter(tsData, Date >= as.Date("2018-12-25"))$n
         MAPE = mean(abs(predicted-obs)/obs)
         MAE = mean(abs(predicted-obs))
         cat("MAE: ", MAE, "\nMAPE: ", MAPE)
```

```
MAE:  10.38382
MAPE:  0.21126
```

0.0.27 Outliers are holidays, what types of crime are being commited on holidays, do holidays have less crime?

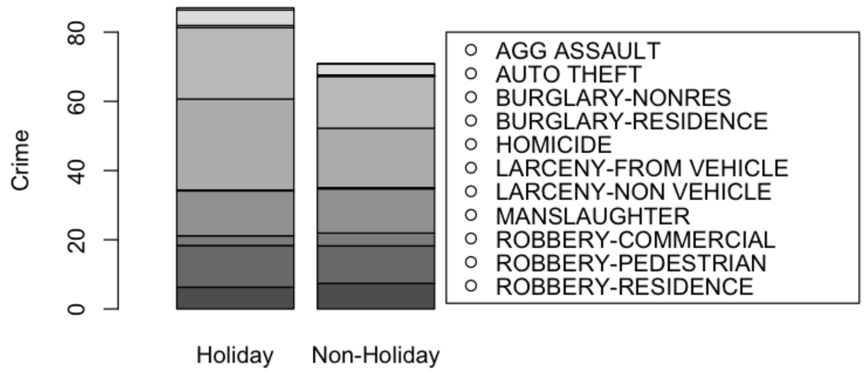
```
In [36]: library(reshape2)
         library(timeDate)
         library(chron)
         CrimeData <- read.csv("COBRA-2009-2018.csv", header=T)
         CrimeData <- select(CrimeData, c(Occur.Date, UCR.Literal))
         names(CrimeData) <- c("Date", "Crime")
         basic_sum = count(CrimeData, Date, Crime)
         names(basic_sum) <- c("Date", "Crime","Count")
         transposed = dcast(basic_sum, Date ~ Crime, value.var = "Count")
         transposed <- filter(transposed, as.Date(Date) >= as.Date("2009-01-01"))
```

```
Attaching package: chron

The following objects are masked from package:lubridate:

    days, hours, minutes, seconds, years
```

```
In [37]: hlist <- c("USChristmasDay","USNewYearsDay")
         myholidays  <- dates(as.character(holiday(2009:2018,hlist)),format="Y-M-D")

In [38]: transposed$Holiday <- is.holiday(as.Date(transposed$Date),myholidays)
         transposed[is.na(transposed)] <- 0
         transposed$Total <- rowSums(transposed[,c(2,3,4,5,6,7,8,9,10,11,12)])
         aggdata <- aggregate(transposed, by=list(Holiday = transposed$Holiday), FUN=mean, na.
```

```
Warning message in mean.default(X[[i]], ...):
argument is not numeric or logical: returning NAWarning message in mean.default(X[[i]], ...):
argument is not numeric or logical: returning NA
```

```
In [67]: ## Plot holidays vs non-holidays
         options(repr.plot.width=8, repr.plot.height=4)
         barplot(t(aggdata)[-c(1, 2, 14,15),],names.arg=c("Holiday","Non-Holiday"), width=10,yl
         legend(x=25,y=80,legend=rownames(t(aggdata)[-c(1, 2, 14,15),]),y.intersp=2,pch=1)
```


Crime
0
20
40
60
80
Holiday
Non-Holiday
AGG ASSAULT
AUTO THEFT
BURGLARY-NONRES
BURGLARY-RESIDENCE
HOMICIDE
LARCENY-FROM VEHICLE
LARCENY-NON VEHICLE
MANSLAUGHTER
ROBBERY-COMMERCIAL
ROBBERY-PEDESTRIAN
ROBBERY-RESIDENCE

```
In [ ]:
```