Toulouse School of Economics

M2 Nonparametric Methods

Take Home Exam

Andrew Boomer

December 22, 2020

Table of Contents

1 Smoothing Splines **2**

1.1 Overview 2

1.2 Cross Validation 4

1.3 Generalized Cross Validation 5

2 Types of Data **5**

2.1 Connection to Spectral Analysis 6

2.2 Cyclic Smoothing Splines 7

3 Dataset **7**

4 Methodology **7**

5 Analysis and Estimation **8**

A Figures and Graphs **12**

B R Code **16**

1 Smoothing Splines

1.1 Overview

The smoothing spline estimator is a type of penalized least squares nonparametric estimator. For a set of observations $i = 1, \cdots, n$, the smoothing spline converts the standard OLS functional form:

$$Y_i = \beta X_i + \epsilon_i$$

into a generalized specification, with an unknown functional form denoted by f. The assumption we make on f is that is has a continuous first and second derivate. Y_i can therefore be expressed as:

$$Y_i = f(x_i) + \epsilon_i$$

The least sqaures minimization of this equation would be similar to ordinary least squares:

$$\sum_{i=1}^{n} [Y_i - f(x_i)]^2$$

[2] calls this aspect the "fidelity" to the data. If this were the only aspect of the minimization for the smoothing spline, then with no functional constraints on $f(x)$ (besides continuous first and second derivates), the solution to the minimization would be to interpolate each data point (with cubic polynomials to ensure differentiability), leading to a perfect fit, with $\epsilon_i = 0 \quad \forall i$. While this technically minimizes the squared errors, this does not constitute actual statistical anylsis.

The smoothing spline deals with this overfitting issue in a similar fashion to how the ridge regression penalizes large parameters. However, where the ridge regression penalized having large paramters, the smoothing spline penalizes roughness in f. This mathematically translates to penalizing the squared second derivate. The second derivate acts a measure of roughness since the second derivative is how much the rate of change is changing. For example, if f transitions from decreasing to increasing, this would be a positive second derivative, and the opposite change would be a negative second derivative. Squaring the second derivate places an even further penalty to select for fewer transitions in f.

The full minimization function for the smoothing spline is therefore:

$$\sum_{i=1}^{n}[Y_i - f(x_i)]^2 + \lambda \int_a^b [f''(x)]^2 dx$$

The λ parameter governs the level of penalty attributed to the roughness of the data, with a smaller λ allowing f to more closely fit the data, and the smoothing spline approaching a straight line as $\lambda \to \infty$.

In terms of implemenation, the smoothing spline can be expressed as a penalized regression spline, using a cubic spline basis. The smoothing penalty will have an additional effect beyond the constraint of the dimension of the basis when the dimension of the cubic basis is sufficiently large. In practice, while a cubic spline would place some number of knots less than the number of data points throughout the range of data, a smoothing spline would allow the number of knots to be equal to the number of data points. This would translate to allowing the λ penalty to constrain the wigglyness of f, rather than dimension of the the basis of the cubic spline.

The minimization problem outlined above in terms of f can be expressed in a quadratic form in terms of a cubic spline basis $\mathbf{X}$:

$$\sum_{i=1}^{n}[Y_i - f(x_i)]^2 + \lambda \int_a^b [f''(x)]^2 dx = \sum_{i=1}^{n}[Y_i - X'\beta]^2 + \lambda\beta' S\beta$$

Since this minimization problem is quadratic, there is a closed-form algebraic solution:

$$\hat{\beta} = (X^{'}X + \lambda S)^{-1}X^{'}y$$

Here, we can see that this solution is nearly identical to the ridge regression minimization solution, except that for the ridge regression, in place of the S matrix, we use the identity matrix, I.

1.2 Cross Validation

While the parameters on the cubic spline basis used can be solved for algebraically given a choice for λ, we also need a way to determine the best choice for λ. If there exists a true function f, then we would like to minimize the squared differences between our estimator $\hat{f}$ and f for all points in the dataset. This is known as the predictive sqaured error, PSE.

$$PSE(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left[\hat{f}_{\lambda}(x_i) - f(x_i)\right]^2$$

Since we cannot know the true functional form f, we can use the large sample property that as $n \to \infty, \hat{f}_{\lambda}^{-i}(x_i) \to \hat{f}_{\lambda}(x_i)$. With this, we can arrive at the leave-one-out cross validation estimator of PSE:

$$\widehat{PSE}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left[\hat{f}_{\lambda}^{-i}(x_i) - y_i\right]^2$$

This can be shown to be independent of the leave-one-out estimator $\hat{f}_{\lambda}^{-i}$, with:

$$\widehat{PSE}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_{i,\lambda}}{1 - S_{ii}}\right)^2$$

Where S_{ii} is the i^{th} diagonal of the smoothing matrix S used in the representation of the second derivative in a cubic spline basis.

1.3 Generalized Cross Validation

The leave-one-out cross validation score is "generalized" by using the average of all the diagonals in the S matrix, rather than each diagonal within the sum. The generalized cross validation score is then therefore:

$$GCV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_{i,\lambda}}{1 - \frac{tr[S]}{n}}\right)^2$$

The generalized cross validation score can also be shown to minimize the squared distance between $\hat{f}$ and f as $n \to \infty$. Additionally, it is computationally easier to calculate the generalized cross validation than the ordinary cross validation score.

2 Types of Data

As with any nonparametric estimator, the smoothing spline is a way to model the relationship between variables. This type of estimator lends itself naturally to cross sectional data, but it can be used in time series data as well, with some further caveats however. In [10], the authors explain that, as with OLS, the standard cross validation methods for the selection of λ are based on the assumption of independent observations. If the observations are autocorrelated, as is generally the case with time series data, then these standard methods fail. The *mgcv* package in R deals with general additive models, and pairs with the *nlme* package which allows for the definition of a non iid correlation structure. With an underlying autocorrelation structure assumed, both λ and the correlation parameters are simultaneously estimated.

However, this approach of using MLE to simultaneously estimate the model and correlation parameters turned out to be computationally expensive to implement, and often was not able to converge depending on the complexity of the correlation structure used. Therefore, I followed the approach for dealing with autocorrelation of the authors in [9]. In this paper, the authors used the generalized additive model framework to perform short term forecasting of hourly electricity load data. While they separately model each hour of the day, they used lagged version of both the dependent and independent variables to address the autocorrelation of the data. Their data and methods are readily applicable to my use case in this report.

2.1 Connection to Spectral Analysis

The penalty parameter λ can also be thought of as a low-pass filter, where a given value of λ specifies the maximum allowed frequency. Lower values of λ would allow higher frequency, or rougher, splines, and higher values would futher restrict higher frequencies.

It can therefore be seen that, for time series data, smoothing splines are intimately tied to the representation of a time series in the frequency domain. In fact, [8] show that a HP-Filter, which is used for time series decomposition, is a type of smoothing spline, as it has been defined thus far. In this context, different values of the λ parameter can be seen as filtering for the different components of a time series. [8] explain that in the limit as $\lambda \to \infty$, the smoothing spline would approach a trend line, where the infinite curvature penalty causes the sum of the squared second derivates to approach zero.

In their original paper describing the HP filter, [7], Hodrick and Prescott derive, for quarterly GDP data, a penalty parameter value of $\lambda = 1600$. This value is derived from contextualizing the smoothing parameter as a ratio of the variance of the long and short phases of the time series. As explained in their overview, [8] discuss the further research that attempted to quantify the appropriate values for the smoothing parameter for time series data measured yearly or monthly. However, [8] argues that a more robust and generalizable approach would be to determine λ strictly from the data rather than on purely theoretical grounds.

2.2 Cyclic Smoothing Splines

To bring this idea of estimating curves of different frequencies back into the world of smoothing splines, we can use the idea of cyclic smoothing splines. A cyclic smoothing spline fits periodic data, such as hourly, daily, or quarterly, and fits a smoothing spline which can be repeated across multiple period windows. A cyclic spline must have the value of the spline, as well as its first and second derivatives, be equal at the breakpoints between two periodic windows.

3 Dataset

The dataset used for this analysis will be hourly traffic count data in the USA. The dataset is dissagregated to the level of a traffic monitoring station, which has a specific location (lat, long) associated with it. Traffic measuring stations can have different purposes, such as for survey purposes, for traffic safety purposes, and others. Additionally, the types of roads can differ, such as rural roads and interstate highways.

In the context of this assignment, the hourly data will be analyzed at the level of a traffic count monitoring station. I have chosen to pick one monitoring station near where I live in the Portland, USA metro area, with no loss of generality given the data driven nature of the exercise.

4 Methodology

This analysis will be approached in the context of attempting to explicitly model the different levels of seasonality in the using the general additive model framework. Traffic count data has several different trend and seasonal components within it, which makes it a well suited dataset for modelling these differing cyclical lengths. Based on a theoretical foundation of traffic behavior, we would expect several periodic levels to be present in this data:

- ▷ Intra-day seasonality, where the peak traffic count occurs in the mid-to-late afternoon, and the daily minima occur in the late night/early morning.
- ▷ Inter-day/weekly seasonality, where different days of the week will show higher/lower traffic counts. For example the average traffic count should be lower on the weekends due to lower commuting volumes than during the week
- ▷ Monthly seasonality, several different factors, including weather, tourism, and others will influence a varying traffic count level throughout the year.
- ▷ While we would expect a multi-year global trend, since this data only spans 1 year, a global trend cannot be separately identified.

The *mgcv* package is well suited to this methodological approach. With this package, a multivariate generalized additive model can be specified. Since this estimation will be the combination of multiple cyclic cubic smoothing splines based on differing time period levels, a multivariate setting is necessary.

As touched on earlier, this is highly related to the idea of the fourier transform, which expresses a time series in terms of a summation of sine and cosine waves within a range of frequencies. Therefore, the goal of this analysis will be to explicitly model the set of frequencies outlined above. To confirm that this theoretical interpretation of the most significant frequencies is correct, a spectral decomposition will be applied to the data to verify these hypotheses.

5 Analysis and Estimation

Figure 1 shows the initial time series plot of the entire dataset, and a randomly selected 2-week subset of the dataset. The daily and weekly cyclical nature is readily apparent.

The cyclical nature postulated in the methodology section is confirmed by the spectral decomposition shown in Figure 2, which shows that the daily cycle is by far the strongest, and obscures identification other cycles.

To approach this estimation, two variations of the GAM will be applied to this data. One will assume that the different periodic time scales have an additive relationship, and the other won't make this assumption.

Additionally, the standard assumption that ϵ_t is a white noise is unlikely to hold for this data, so this assumption will be modified to allow the error terms to follow some kind of $ARMA$ process. The identification of this serial correlation will be initially done in a two-step approach, first fitting the GAM models, then identifying the autocorrelation structure of the residuals using the *auto.arima* function in the $forecast$ package in R, and re-estimating the GAM model with the correct AR and/or MA values.

Therefore, the first model is:

$$TrafficCount_t = f_1(Hour_t) + f_2(Day_t) + f_3(Month_t) + \epsilon_t$$

The cyclic cubic smoothing splines for the daily and weekly periods are shown separately in Figure 3. They are shown in a 3D plot in Figure 4. Given the initial time series plot and spectral decomposition in Figures 1 and 2, these results seem good. Additionally, the adjusted R^2 for this model is 89.5%, and all the spline terms are significant. However, as expected, there is strong serial correlation in the residuals. The fitted vs. true values, residuals, and ACF/PACF of the residuals are plotted in Figure 5.

The ACF and PACF of the residuals indicate that the smoothing splines haven't fully accounted for the autocorrelation of the series at the daily and weekly level. This should make sense as the cyclic aspect forces a repetition of the splines in Figure 3 through the entire dataset. Because of this property, this estimation is missing pertinent information about the actual level of recent values.

To account for this, a few AR terms should be included so that the level of the cyclic smoothing splines can be adjusted to more closely match the stochastic change throughout the full time span. Additionally, the forced additive relationship between the daily and weekly cycles is restrictive and is relaxed in the second model.

The second model, where $T = \{1, 2, 3, 24, 168\}$, is:

$$TrafficCount_t = f_1(Hour_t, Day_t) + f_2(Month_t) + \sum_{j \in T} f_j(TrafficCount_{t-j}) + \epsilon_t$$

The 3D relationship between the daily and weekly cycles and Traffic Count are shown in Figure 6. Additionally the fitted values compared to the true values, the residuals, and the ACF/PACF of the residuals are shown in Figure 7. Inspecting the plot of the residuals, as well as the ACF and PACF graphs, this model appears to be much closer to satisfying the assumption of uncorrelated errors. Running the *auto.arima* function on the residuals returns a model with no additional AR or MA terms, lending further evidence that this is a correct specification.

In this specification, all the spline terms are significant, and the adjusted R^2 has jumped to 98.6%, indicating a much closer fit to the data. This is to be expected given the extra information provided by the lagged values of the dependent variable. Additionally, the GCV score is lower in this model by about a factor of 10.

References

[1] Anestis Antoniadis, Efstathios Paparoditis, and Theofanis Sapatinas. "A Functional Wavelet-Kernel Approach for Continuous-time Prediction". In: *arXiv preprint math/0505172* ().

[2] Dursun Aydın, Memmedağa Memmedli, and Rabia Ece Omay. "Smoothing parameter selection for nonparametric regression using smoothing spline". In: *European Journal of Pure and applied mathematics* 6.2 (2013), pp. 222–238.

[3] Helen XH Bao and Alan TK Wan. "On the use of spline smoothing in estimating hedonic housing price models: empirical evidence using Hong Kong data". In: *Real estate economics* 32.3 (2004), pp. 487–507.

[4] Nicoleta Breaz. "The cross-validation method in the smoothing spline regression". In: *Acta Universitatis Apulensis, Mathematics-Informatics, Proc. of Int. Conf. on Theory and Appl. of Math. and Inf., Alba Iulia*. Vol. 7. 2004, pp. 77–84.

[5] Alfred Greiner. "Estimating penalized spline regressions: Theory and application to economics". In: *Applied Economics Letters* 16.18 (2009), pp. 1831–1835.

[6] Alfred Greiner and Göran Kauermann. "Sustainability of US public debt: Estimating smoothing spline regressions". In: *Economic Modelling* 24.2 (2007), pp. 350–364.

[7] Robert J Hodrick and Edward C Prescott. "Postwar US business cycles: an empirical investigation". In: *Journal of Money, credit, and Banking* (1997), pp. 1–16.

[8] Göran Kauermann, Tatyana Krivobokova, and Willi Semmler. "Filtering time series with penalized splines". In: *Studies in Nonlinear Dynamics & Econometrics* 15.2 (2011).

[9] Amandine Pierrot and Yannig Goude. "Short-term electricity load forecasting with generalized additive models". In: *Proceedings of ISAP power* 2011 (2011).

[10] Yuedong Wang. "Smoothing spline models with correlated random errors". In: *Journal of the American Statistical Association* 93.441 (1998), pp. 341–348.

A Figures and Graphs

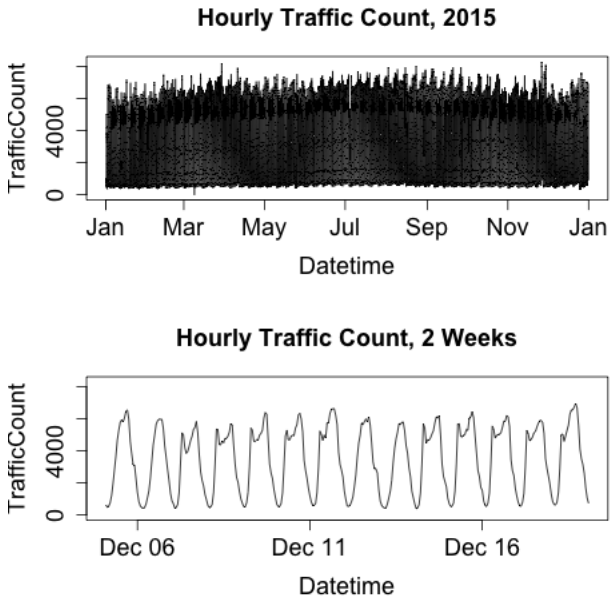

Figure 1

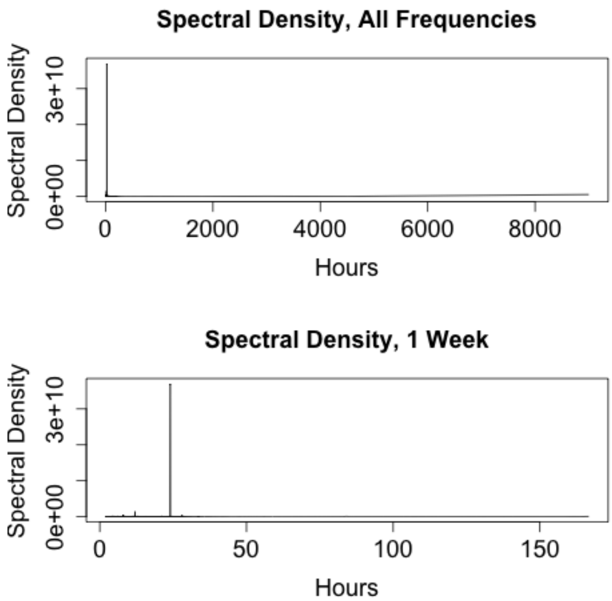

Figure 2

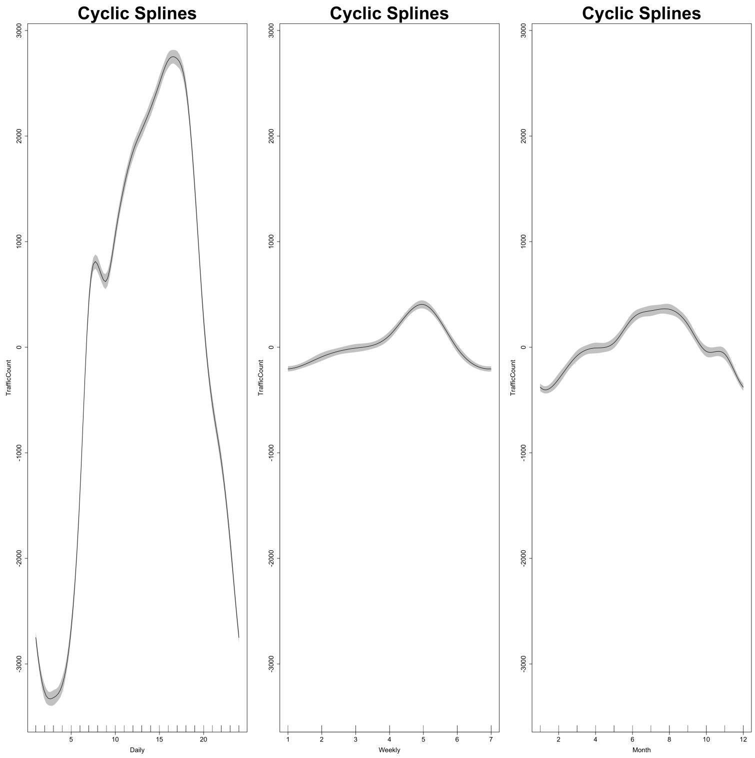

Figure 3

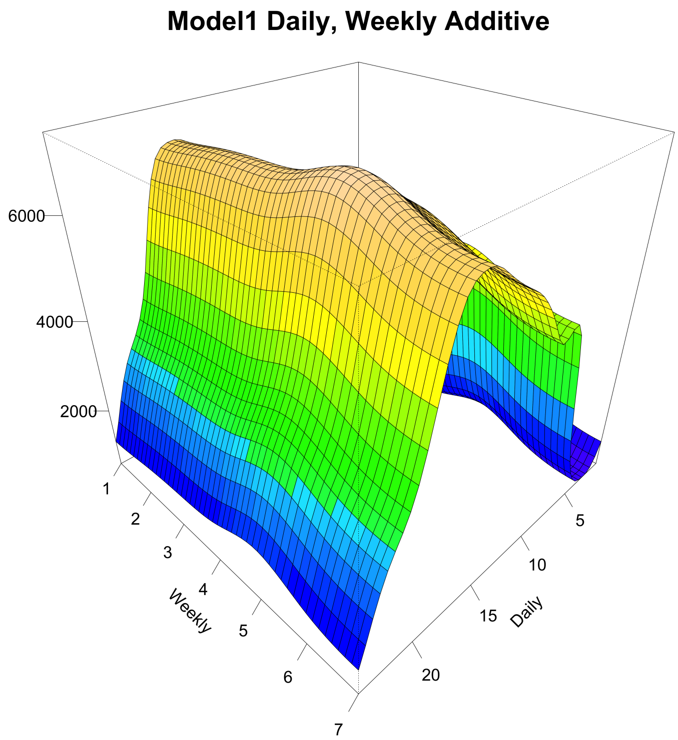

Figure 4

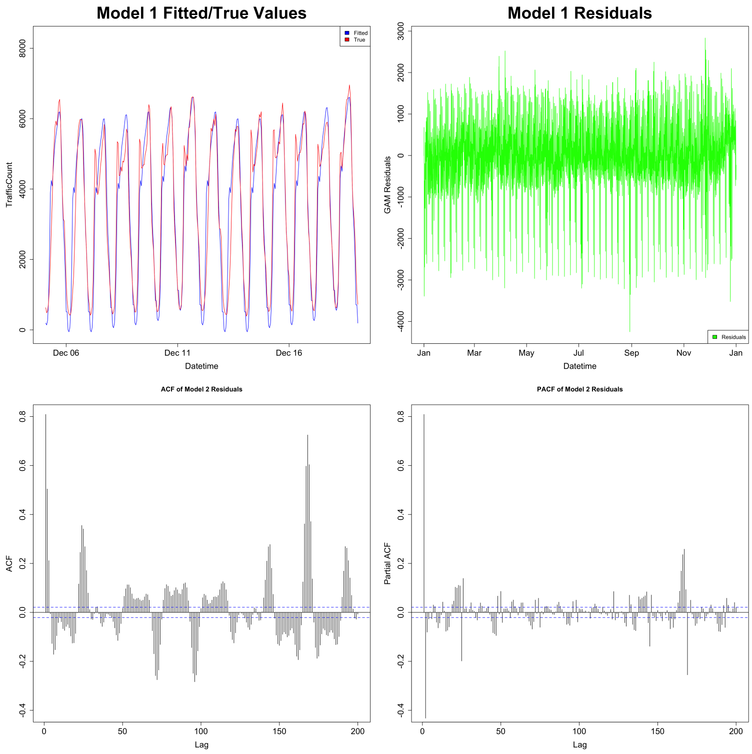

Figure 5

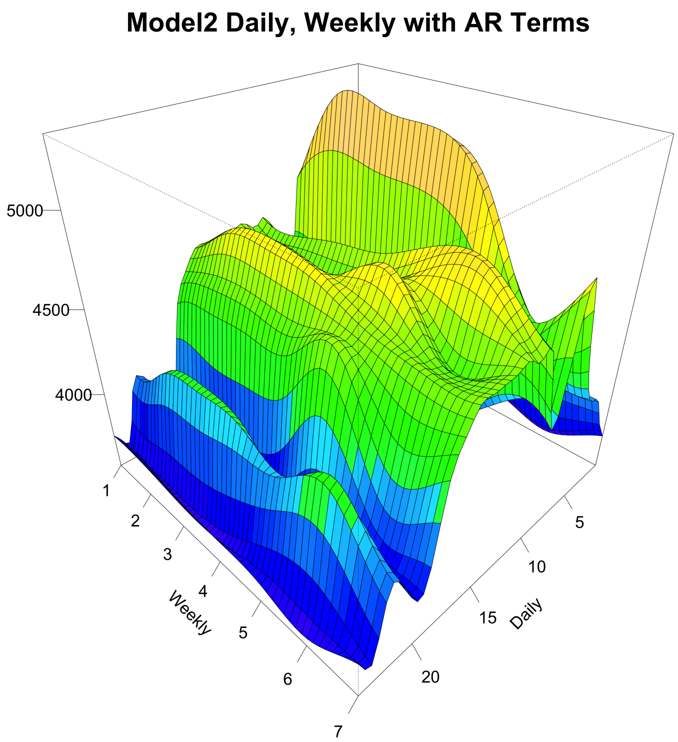

Figure 6

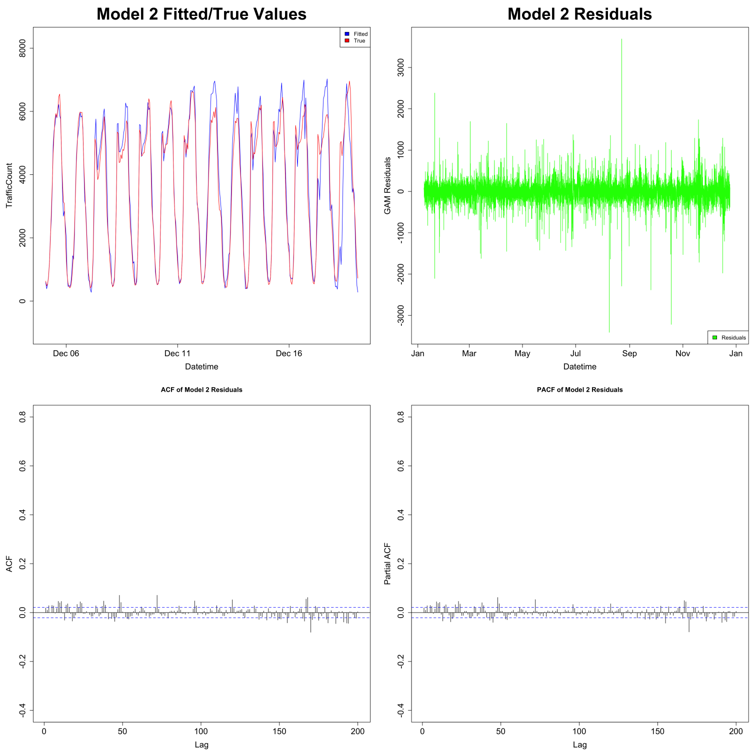
Model 2 Fitted/True Values
Fitted
True
TrafficCount
Datetime
Dec 06
Dec 11
Dec 16
Model 2 Residuals
GAM Residuals
Residuals
Jan
Mar
May
Jul
Sep
Nov
Jan
ACF of Model 2 Residuals
ACF
Lag
PACF of Model 2 Residuals
Partial ACF
Lag

Figure 7

B R Code

```r
#################################################
# Library Importation
#################################################
library(tidyverse)
library(data.table)
library(stats)
library(pspline)
library(mapproj)
library(lubridate)
library(ASSIST)
library(TSA)
library(nlme)
library(bigsplines)
library(splines)
library(mgcv)
library(ggplot2)
library(forecast)

#################################################
# Data Importation
#################################################

data_dir <- paste0(getwd(), "/archive")

stations_file <- paste0(data_dir, "/dot_traffic_stations_2015.txt")

stations <- read.csv(stations_file, sep=',')
```

```r
stations$fips_county_code <- sapply(stations$fips_county_code,
    function(x) {str_pad(x, 3, side='left', pad="0")})
stations$fips_state_code <- sapply(stations$fips_state_code,
    function(x) {str_pad(x, 2, side='left', pad="0")})

df <- read.csv(paste0(data_dir, "/trimmed.csv"), sep=',')
df$station_id <- str_pad(df$station_id, 6, side='left', pad='0')

###############################################
# Data Cleaning and Preparation
###############################################

pdx <- '003011' #Get a station in the Portland metro area

#Aggregate the data by datetime and station_id
data <- aggregate(df$TrafficCount, by=list(df$datetime,
    df$station_id), FUN=sum)
names(data) <- c("datetime", 'station_id', 'TrafficCount')

data <- data[data$station_id == pdx, ] #Filter to chosen station id

data$Daily <- rep(1:24, 365) #Create a list of repeating hours of
    the day
data$Weekly <- wday(data$datetime, week_start=1) #Get the day of the
    week
data$Month <- month(data$datetime) #Get the month number
#Convert the datetime to a posix variable
data$datetime <- as.POSIXct(strptime(data$datetime, "%Y-%m-%d
```

```
    %H:%M:%S"), tz='GMT')

data$Time <- 1:dim(data)[1] #Column for the time index
data$station_name <- "Portland Metro" #Name the station
data$Date <- as.Date(data$datetime) #Get just the date portion of
    the datetime

#Get a set of lags of the Traffic Count variable to handle
    autocorrelation
data$Lag1 <- shift(data$TrafficCount, 1)
data$Lag2 <- shift(data$TrafficCount, 2)
data$Lag3 <- shift(data$TrafficCount, 3)
data$Lag24 <- shift(data$TrafficCount, 24)
data$Lag48 <- shift(data$TrafficCount, 48)
data$Lag168 <- shift(data$TrafficCount, 168)

###############################################
# Exploratory Analysis of Data
###############################################

#Generate a random starting index for a 2 week length for initial
    plot
len <- 24*7*2
start <- runif(1, 169, dim(data)[1])

ymin <- min(data$TrafficCount) #Get minimum of the data
ymax <- max(data$TrafficCount) #Used for defining the y-axis maximum

```

```
png("InitYearPlot.png") #Call the png function in base R
layout(matrix(1:2, nrow=2)) #Make a plotting grid
#Plot the full dataset of the traffic count per hour
plot(data$datetime, data$TrafficCount, type='l', col='black',
    xlab='Datetime', ylab='TrafficCount', ylim=c(0, ymax),
    main="Hourly Traffic Count, 2015",
    cex.lab=1.5, cex.axis=1.5, cex.main=1.5)
#Plot just the randomly chosen two week subset
plot(data[start:(len + start),]$datetime,
    data[start:(len + start),]$TrafficCount, type='l', col='black',
    xlab='Datetime', ylab='TrafficCount', ylim=c(0, ymax),
    main="Hourly Traffic Count, 2 Weeks",
    cex.lab=1.5, cex.axis=1.5, cex.main=1.5)
dev.off() #Store the plot as a png file

#Get the periodogram dataframe, showing the spectral density at
   frequencies
ft <- periodogram(data$TrafficCount, plot=FALSE)
ft <- data.frame(ft$freq, ft$spec) #Put into a dataframe
names(ft) <- c("freq", 'spec') #Rename the columns
ft$Hours <- 1/ft$freq #The inverse of the frequency is the number of
   hours

png("InitSpec.png") #Develop a plot of the spectral density
layout(matrix(1:2, nrow=2, ncol=1)) #Two column plotting grid
#Plot the full frequency series of the spectral density
plot(ft$Hours, ft$spec, type='l', col='black', xlab='Hours',
    ylab='Spectral Density', main="Spectral Density, All
```

```
    Frequencies",
    cex.lab=1.5, cex.axis=1.5, cex.main=1.5)
#Plot just the first week, 168 hours of the spectral density
plot(ft[ft$Hours <= 168,]$Hours, ft[ft$Hours <= 168,]$spec, type='l',
    col='black', xlab='Hours', ylab='Spectral Density',
    main="Spectral Density, 1 Week", cex.lab=1.5, cex.axis=1.5,
      cex.main=1.5)
dev.off() #Store as a png

###############################################
# Generalized Additive Model Estimation
###############################################

    ######################################################
    # First GAM model, testing chosen variables groupings
    ######################################################

#Specifying cyclic cubic splines with the correct periods, also
   constraining
# them to have an additive relationship.
mod1 <- gam(TrafficCount ~ s(Daily, bs='cc', k=24) + s(Weekly,
   bs='cc', k=7) +
    s(Month, bs='cc', k=12),
    data = data, family = gaussian)

png("Mod1_2D.png", units='mm', res=300)
layout(matrix(1:3, nrow = 1)) #2 column plotting grid
plot(mod1, shade = TRUE, ylab='TrafficCount', cex.lab=1.5,
```

```
  cex.axis=1.5,
   cex.main=4, main='Cyclic Splines') #Plot the model
dev.off()

png("Mod1_3D.png", units='mm', res=300)
vis.gam(mod1, view=c('Daily', 'Weekly'), n.grid = 50, theta = 135,
  phi = 32, zlab = "", ticktype = "detailed", color = "topo",
   main="Model1 Daily, Weekly Additive",
   cex.lab=2.5, cex.axis=2.5, cex.main=4)
dev.off()

#Get the residuals, fitted values
res_1 <- mod1$residuals
mod1_fit <- mod1$fitted.values
min1 <- min(ymin, min(mod1_fit)) #Get min and max for plotting
max1 <- max(ymax, max(mod1_fit))

png("Model1.png", units='mm', res=300) #Store the model
layout(matrix(1:4, nrow = 2, ncol=2)) #Get 2x2 grid for plotting
#Plot a random subset of the fitted values and the true values
plot(data[start:(start+len),]$datetime,
   mod1_fit[start:(start+len)], ylab='TrafficCount',
     xlab='Datetime',
   ylim=c(min1, max1), col='blue', type='l', main="Model 1
     Fitted/True Values",
   cex.lab=1.5, cex.axis=1.5, cex.main=3)
lines(data[start:(start+len),]$datetime,
  data[start:(start+len),]$TrafficCount, ylab='TrafficCount',
```

```
   xlab='Datetime', ylim=c(min1, max1), col='red')
#Get the legend
legend('topright', c("Fitted", 'True'), fill=c("blue", 'red'))
#Plot the ACF up to 200 lags
plot(acf(res_1, lag=200, plot=FALSE), ylim=c(-0.4, 0.8),
    main="ACF of Model 2 Residuals", cex.lab=1.5, cex.axis=1.5,
       cex.main=1.5)

#Plot the random subset for the true values
plot(data$datetime, res_1,
    ylab='GAM Residuals', xlab='Datetime', ylim=c(min(res_1),
       max(res_1)),
    col='green', type='l', main="Model 1 Residuals",
    cex.lab=1.5, cex.axis=1.5, cex.main=3)
legend('bottomright', 'Residuals', fill='green')

#Plot the PACF values for lags up to 200
plot(pacf(res_1, lag=200, plot=FALSE), ylim=c(-0.4, 0.8),
    main="PACF of Model 2 Residuals",
    cex.lab=1.5, cex.axis=1.5, cex.main=1.5)
dev.off()

#See what AR and MA terms should be included based on the residuals
arima1 <- auto.arima(mod1$residuals, stationary=TRUE, seasonal=TRUE)

    #####################################################
    # Second GAM model relaxing additive model assumption, including
       AR terms
```

```
    ######################################################

#Add AR terms to model the autocorrelation present in the time series
mod2 <- gam(TrafficCount ~ t2(Daily, Weekly, bs=c('cc', 'cc'),
   k=c(24, 7)) +
            t2(Month, bs='cs') +
            t2(Lag1, bs='cs') + t2(Lag2, bs='cs') + t2(Lag3,
               bs='cs') +
            t2(Lag24, bs='cs') + t2(Lag168, bs='cs'),
            data = data, family=gaussian)

png("Mod2_3D.png", units='mm', res=300)
vis.gam(mod2, view=c('Daily', 'Weekly'), n.grid = 50, theta = 135,
    phi = 32, zlab = "", ticktype = "detailed", color = "topo",
    main='Model2 Daily, Weekly with AR Terms',
    cex.lab=2.5, cex.axis=2.5, cex.main=4)
dev.off()

#Get the fitted values and residuals from the second model
mod2_fit <- mod2$fitted.values
res_2 <- mod2$residuals
min2 <- min(ymin, min(mod2_fit))
max2 <- max(ymax, max(mod2_fit))

png("Model2.png", units='mm', res=300) #Store the plot in a png file
layout(matrix(1:4, nrow = 2)) #Layout a 2x2 grid for plotting
#Plot the fitted and true values for the chosen subset of data
plot(data[start:(start+len),]$datetime, mod2_fit[start:(start+len)],
```

```
    ylab='TrafficCount', xlab='Datetime', ylim=c(min2, max2),
       col='blue',
    type='l', main="Model 2 Fitted/True Values",
    cex.lab=1.5, cex.axis=1.5, cex.main=3)
lines(data[start:(start+len),]$datetime,
   data[start:(start+len),]$TrafficCount,
    ylab='TrafficCount', xlab='Datetime', ylim=c(min2, max2),
       col='red')
legend('topright', c("Fitted", 'True'), fill=c("blue", 'red'))
#Plot the ACF of the residuals of the 2nd model
plot(acf(res_2, lag=200, plot=FALSE), ylim=c(-0.4, 0.8),
    main="ACF of Model 2 Residuals", cex.lab=1.5, cex.axis=1.5,
       cex.main=5)
#Plot the residuals of the 2nd model for the full dataset
plot(data[169:nrow(data),]$datetime, res_2[169:nrow(data)],
    ylab='GAM Residuals', xlab='Datetime', ylim=c(min(res_2),
       max(res_2)),
    col='green', type='l', main="Model 2 Residuals",
    cex.lab=1.5, cex.axis=1.5, cex.main=3)
legend('bottomright', 'Residuals', fill='green')
#Plot the PACF of the residuals of the 2nd model up to 200 lags
plot(pacf(res_2, lag=200, plot=FALSE), ylim=c(-0.4, 0.8),
    main="PACF of Model 2 Residuals", cex.lab=1.5, cex.axis=1.5,
       cex.main=5)
dev.off()

auto.arima(mod2$residuals, stationary=TRUE, seasonal=TRUE)
```