# Web Traffic Forecasting

Time Series Analysis

**Shree Karimkolikuzhiyil**; Programming; Statistics, M.S. (Distance); shreejesh@tamu.edu
**Jingcheng Xia**; Computations; Computer Science, B.S. (On-Campus); sixtyfour64@tamu.edu
**Jackson Smith**; Analysis; Statistics, M.S. (Distance); jackson.t.smith@tamu.edu
**Samuel Burge**; Writing; Statistics, M.S. (Distance); samuelburge@tamu.edu
**Max Kutschinski**; Theory; Statistics, M.S. (Distance); mwk556@tamu.edu

---

## Introduction and Motivation

The objective of this analysis is to forecast daily unique visitors to an academic website containing lecture notes and supplemental material related to statistics. Predicting website traffic allows IT departments to manage project throughput and prioritize maintenance and enhancements to website functionality and effectively allocate web server resources. Web traffic is also a key indicator of customer growth and expansion, as well as sustaining recurring customers and ingrained growth. The details provided by web traffic throughput reports contain many metrics, including page loads, returning visitors, and unique visits, each of which conveys a different picture and set of information for an organization. As well, having a picture of expected throughput and confirming (or denying) expectations with reality allows a business to understand unexpected growth and/or unexpected decay in business development.

The data contains five years of daily time series data of user visits. There are four features in the data set, which include daily counts for the number of page loads, first-time visitors, returning visitors, and unique visitors.[1] An initial plot of the data shows strong seasonality and volatility, but doesn't appear to have any discernible trend or cyclical behavior. An explanation for this could be due to the nature of the website. Students would likely be the largest share of users for a website of this nature, and the seasonality seems associated with the academic calendar typically seen at academic institutions.

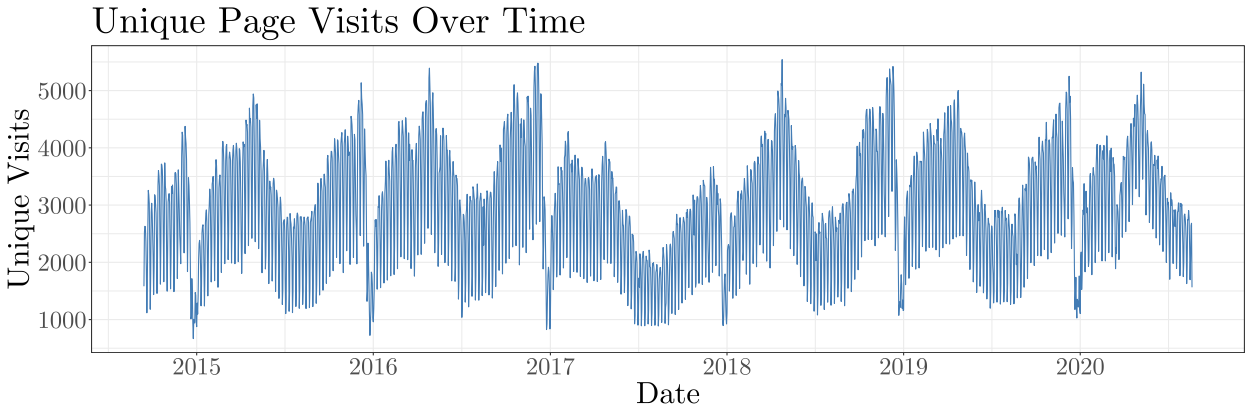



Figure 1

[1] A visit is defined as a stream of hits on one or more pages on the site on a given day by the same user within a 6-hour window, identified by the IP address of the specific device. Returning visitors are identified through allowed cookies on a user's device, and the total number of returning and first-time visitors is, by definition, the number of unique visitors.

### Stationarity

Stationarity is a common assumption underlying many time series procedures. As such, it is important to assess the level of stationarity prior to modeling and make the appropriate adjustments if necessary.

A stationary time series is one whose properties do not depend on the time at which the series is observed. More specifically,

(i) *the mean value function* $\mu_t = E(x_t)$ *is constant and does not depend on time* $t$

(ii) *the autocovariance function* $\gamma(s,t) = cov(x_s, x_t) = E[(x_s - \mu_s)(x_t - \mu_t)]$ *depends on times* $s$ *and* $t$ *only though their lagged difference.*

The strong seasonality that is apparent in Figure 1 is indicative of non-stationarity, since seasonality will affect the value of the time series at different times. Seasonality is defined as a recurring pattern at a fixed and known frequency based on the time of the year, week, or day.

Figures 2 and 3 aim to identify the types of seasonality present in the data. Figure 2 plots a subset of the first several weeks and indicates that there exists weekly seasonality, whereas Figure 3 uses locally weighted scatterplot smoothers (Lowess) to emphasize the inherent yearly seasonality.

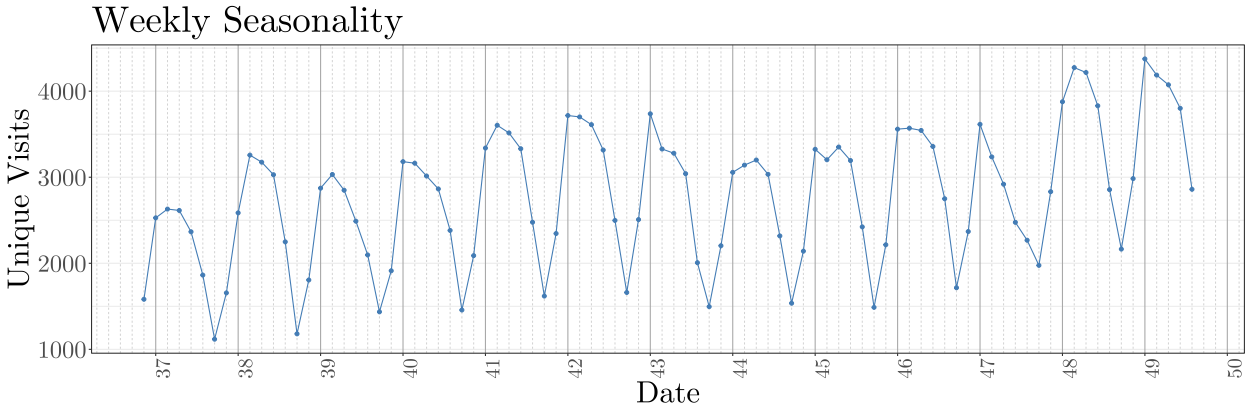



Figure 2: Sample of weekly page visits

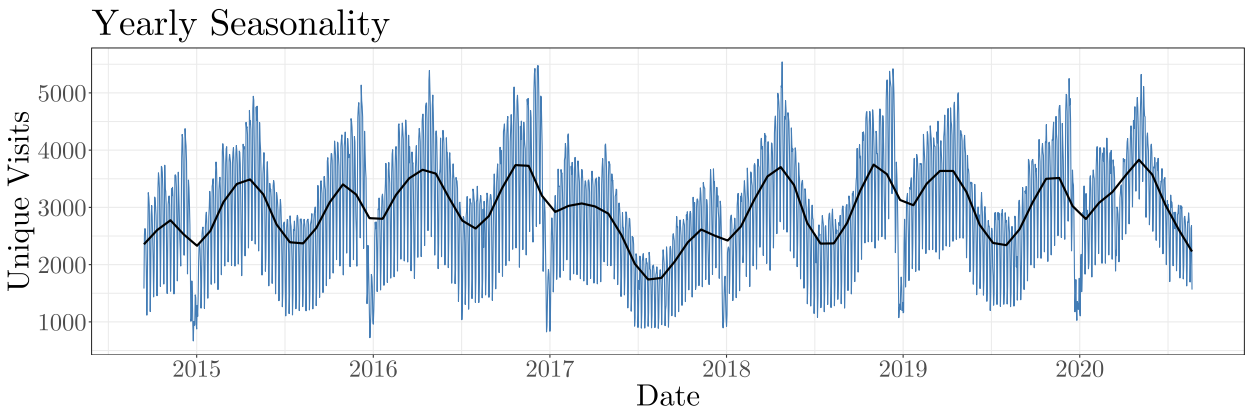



Figure 3: Smoothing via Lowess

A popular approach in addressing non-stationarity due to seasonalities is to eliminate these effects via seasonal differencing. The seasonal difference of a time series is the series of changes from one season to the next, which is defined as folllows:

$$\nabla x_t = x_t - x_{t-m} \tag{1}$$

Hence, yearly and weekly seasonalities will be handled by computing the lag 365 and lag 7 seasonal difference, respectively.

$$\nabla x_t^* = (x_t - x_{t-365}) - x_{t-7} \tag{2}$$

Time plots of our series at different levels of differencing are displayed in Figure 4. The differenced series appears to be stationary with constant mean and variance.

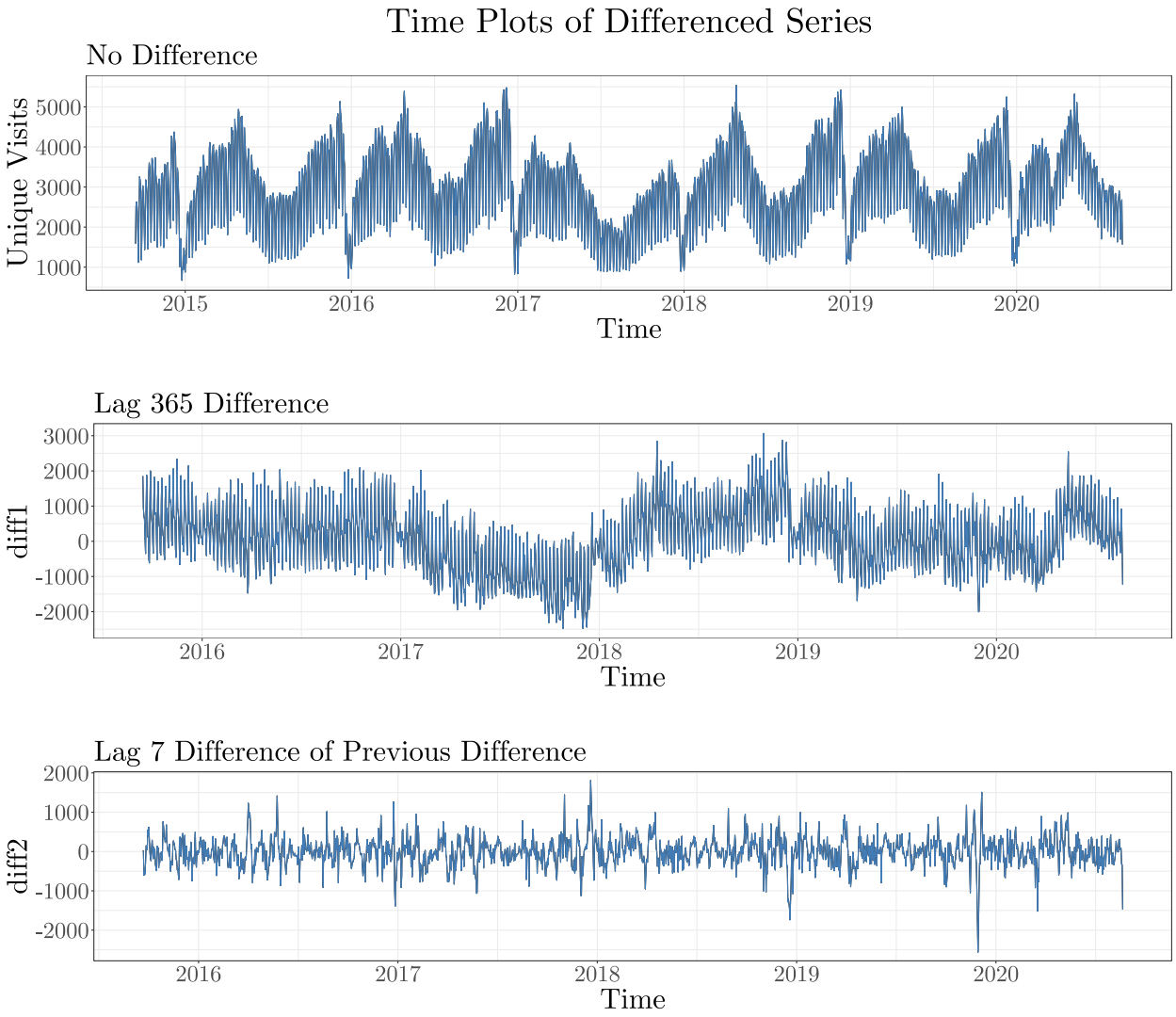



Figure 4: Time plots of differenced series

The ACF and PACF of the differenced series $\nabla x_t^*$ are displayed in Figure 5. Neither the ACF nor the PACF seems to cut off after a certain lag, which would be indicative of an AR or MA process. Rather, both of them appear to tail off over time.

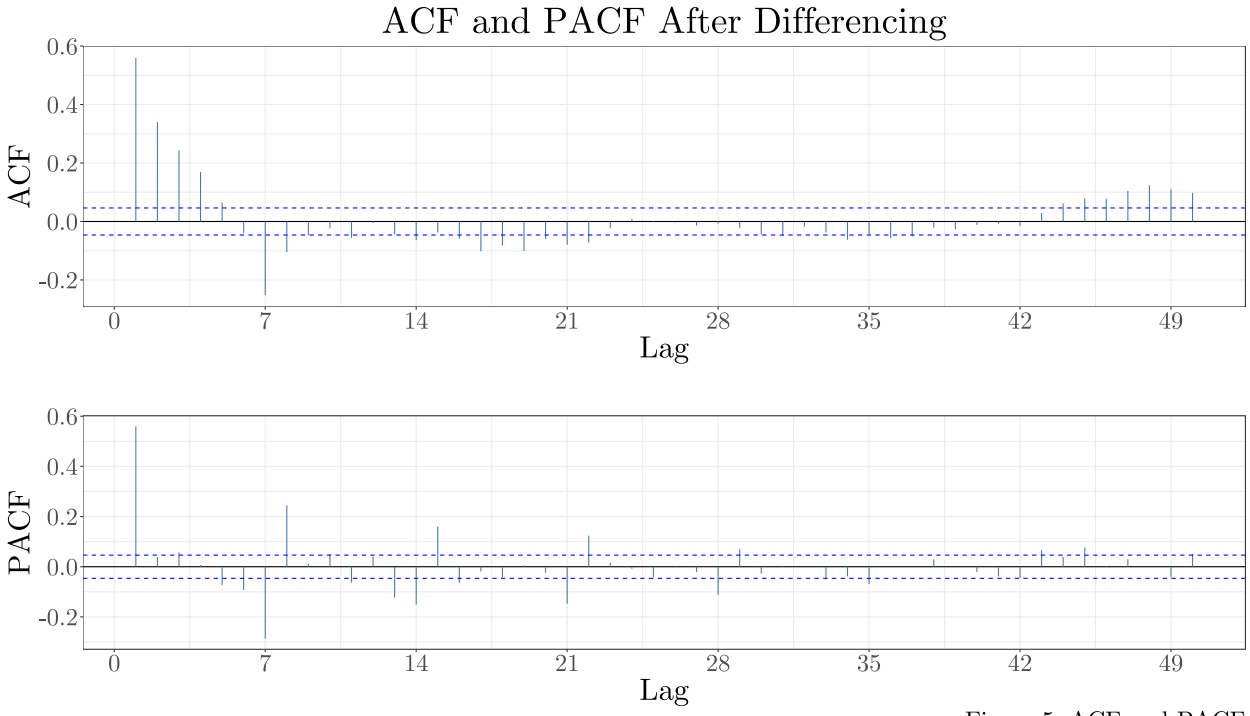



Figure 5: ACF and PACF

## Modeling

Both ACF and PACF show a slow decay indicating that an ARMA(2,2) model could be appropriate for the series. An auto regressive moving average of order p and q (ARMA(p,q)) model is defined in Eq(3):

$$x_t = \alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 \omega_{t-1} + \cdots + \theta_p \omega_{t-q} \tag{3}$$

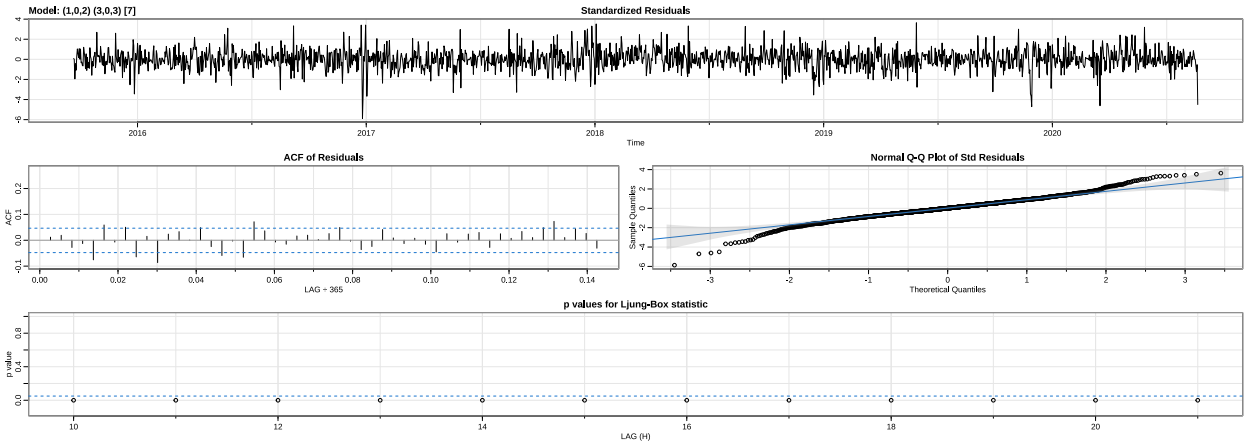
where $\phi_p \neq 0, \theta_p \neq 0, \sigma_w^2 > 0$, and the model is causal and invertible.

```
## initial  value 5.929372
## iter   2 value 5.765890
## iter   3 value 5.648125
## iter   4 value 5.629152
## iter   5 value 5.620266
## iter   6 value 5.615194
## iter   7 value 5.604479
## iter   8 value 5.594269
## iter   9 value 5.587414
## iter  10 value 5.573725
```

```
## iter  11 value 5.571981
## iter  12 value 5.570533
## iter  13 value 5.570046
## iter  14 value 5.569784
## iter  15 value 5.569749
## iter  16 value 5.569743
## iter  17 value 5.569738
## iter  18 value 5.569686
## iter  19 value 5.569609
## iter  20 value 5.569485
## iter  21 value 5.569421
## iter  22 value 5.569382
## iter  23 value 5.569378
## iter  24 value 5.569374
## iter  25 value 5.569363
## iter  26 value 5.569362
## iter  27 value 5.569361
## iter  28 value 5.569360
## iter  29 value 5.569360
## iter  30 value 5.569360
## iter  31 value 5.569359
## iter  32 value 5.569359
## iter  33 value 5.569357
## iter  34 value 5.569355
## iter  35 value 5.569353
## iter  36 value 5.569353
## iter  36 value 5.569352
## iter  36 value 5.569352
## final  value 5.569352
## converged
## initial  value 5.569379
## iter   2 value 5.569374
## iter   3 value 5.569360
## iter   4 value 5.569355
## iter   5 value 5.569350
## iter   6 value 5.569321
## iter   7 value 5.569291
## iter   8 value 5.569259
## iter   9 value 5.569222
## iter  10 value 5.569180
## iter  10 value 5.569180
## final  value 5.569180
## converged

## Warning in sqrt(diag(fitit$var.coef)): NaNs produced

## Warning in sqrt(diag(fitit$var.coef)): NaNs produced
```




```
##                    Length Class  Mode
## fit                14     Arima  list
## degrees_of_freedom  1     -none- numeric
## ttable             40     -none- numeric
## AIC                 1     -none- numeric
## AICc                1     -none- numeric
## BIC                 1     -none- numeric
```

The parameters of the ARMA(2,2) model are estimated via conditional least squares and are displayed in Eq(4).

$$\hat{x}_t = 0.44x_{t-1} + 0.13x_{t-2} + w_t + 0.05\omega_{t-1} - 0.09\omega_{t-2} \tag{4}$$

Figure 6 displays of plot of the residuals of the fitted ARMA(2,2) model. The residuals seem to behave like white noise, being centered around 0 with constant mean and variance.

Table 1 contains the roots of the model's polynomials. Since they appear to be different from each other by a reasonable margin, we can conclude that there is no parameter redundancy in the model.

The fitted values of the ARIMA(2,2) model are plotted against the actual values of the differenced series $\nabla x_t^*$ in Figure 7.

```
m=dshw(subset(tsData[,5],end=length(tsData[,5])-7),h=7)


auto.arima(tsData[,5], seasonal=T)
```

When there are long seasonal periods, a dynamic regression with Fourier terms is often better than other models we have considered in this book. Seasonal versions of ARIMA and ETS models are designed for shorter periods such as 12 for monthly data or 4 for quarterly data.

```
fit <- auto.arima(cafe04, xreg = fourier(cafe04, K = i),
    seasonal = FALSE, lambda = 0)
  plots[[i]] <- autoplot(forecast(fit,
      xreg=fourier(cafe04, K=i, h=24))) +
    xlab(paste("K=",i,"   AICC=",round(fit[["aicc"]],2))) +
    ylab("") + ylim(1.5,4.7)
```

```
m=mstl(tsData[,5])
plot(m)

# train test split
UniqueVisits = tsData[,5]
training <- subset(UniqueVisits, end=length(UniqueVisits)-365)
test <- subset(UniqueVisits, start=length(UniqueVisits)-364)

# baseline

web.train <- Arima(training, order=c(2,1,1),
  seasonal=c(0,1,2), lambda=0)
cafe.train %>%
  forecast(h=60) %>%
  autoplot() + autolayer(test)

snaive(test,h=7)
```

```
tsData[,5] %>%  stlf() %>%
  autoplot() + xlab("Week")
```

```
fit <- auto.arima(tsData[,5], seasonal=FALSE,
         xreg=fourier(tsData[,5], K=c(3,3))
fit %>%
  forecast(xreg=fourier(tsData[,5], K=c(10,10))) %>%
  autoplot() +
    ylab("Call volume") + xlab("Weeks")
```