Time Series Analysis Using R's Statistical Functions

Stat 133 Post #1, Fall 2017

Edward Yang

I. Introduction

A Simple Model: Time

In Statistics 133, we have been analyzing various data sets through the use of well-known R packages, such as `dplyr` for data organization and `ggplot2` for graphical visualization. In nearly all of these data sets, such as the NBA players' salaries in 2017, time is not considered as a variable because we are analyzing only one particular data set in one period of time; however, we know that time is one of the most influential variables in data sets that are studied longitudinally.

A time series is a sequence of data points that are displayed in order. The choice of this display, such as a graph (similar to a line chart) or a chain of lists, varies on what topic and type of data is being studied. As a result, the purpose of this post is to use built-in R functions and installed packages that we have yet learned as methods to illustrate the potential effects of time in data sets and analyze this data in a different way – through the use of regression models to forecast the future with appropriate extrapolation of time.

Motivation: Why Time Series?

As an Economics major, I was curious to learn how statistics can be applied in a setting where there is time inconsistency for consumer preferences and how that can change decisions. In the first two months of our statistics class, we have not studied time as a variable, but, intuitively, all of us know that time is crucial for making appropriate conclusions about **why** a particular data set is behaving a certain way and **how** statistical models can be applied to a specific time setting. I plan on taking Statistics 153, which is a Times Series class, which can easily be applied to a variety of different fields, from computer science/mathematics (Markov chains) to physics/biology (human error).

During this project, I reminded myself of the following questions:

1. What is the purpose of studying time as a variable?
2. How can we use time to support our data sets?
3. Why is time not applicable in certain statistical models?

Enough babbling; let's get onto learning how to create time series.

II. Getting Started: Creating Time Series Objects and Plotting

Example: Lake Huron

As most of the class has not learned any form of times series analysis in R before, let's consider a simple (and easily comprehensible) example with Lake Huron, one of the Great Lakes that borders Michigan. For this example, we will be using a bit of what we learned in Statistics 133 (just so you understand where we are coming from) and then we will quickly proceed to a much more in-depth and meaningful example.

Reference 1: Lake Huron - https://vincentarelbundock.github.io/Rdatasets/datasets.html

The data of Lake Huron has been downloaded in its .csv form from the link above (Reference 1). The data contains annual height level, in feet, for Lake Huron from 1875 to 1972 (98 years). A copy of it has been saved as as a .csv file named `huron.csv` in the `data` folder.

```
huron <- read.csv('../data/huron.csv')
head(huron, 10) # displays the first 10 years of heights
```

```
##      X time LakeHuron
## 1    1 1875    580.38
## 2    2 1876    581.86
## 3    3 1877    580.97
## 4    4 1878    580.80
## 5    5 1879    579.79
## 6    6 1880    580.39
## 7    7 1881    580.42
## 8    8 1882    580.82
## 9    9 1883    581.40
## 10 10 1884    581.32
```

Let's make sure there are 98 rows to correspond to each of the 98 years.

```
dim(huron) # shows that there are indeed 98 rows for the 98 years
```

```
## [1] 98  3
```

Now, we will begin using the built-in function `ts()` to store the data as a time series object. `ts()` takes a vector or matrix and returns a time series object.

```
# creating time series object starting at year 1875
huron_ts <- ts(huron$LakeHuron, start = c(1875, 1))
huron_ts
```

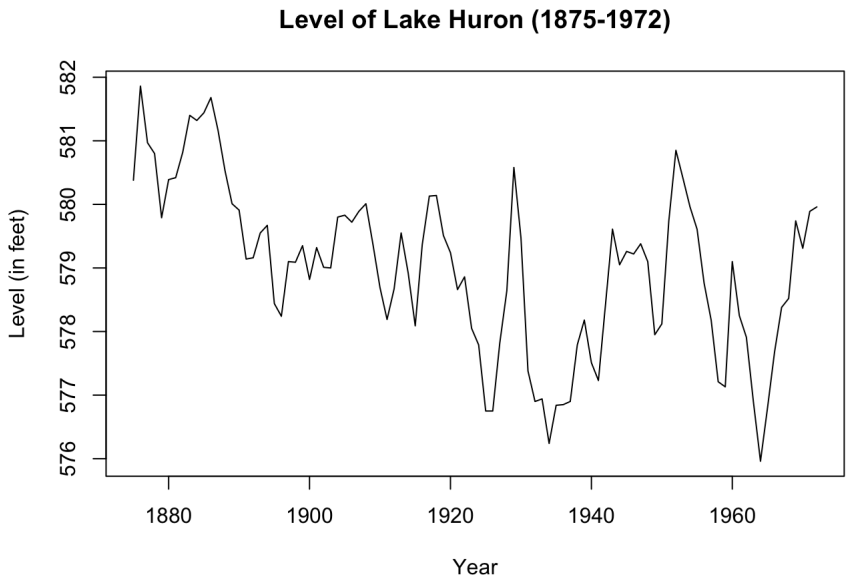
```
## Time Series:
## Start = 1875
## End = 1972
## Frequency = 1
##  [1] 580.38 581.86 580.97 580.80 579.79 580.39 580.42 580.82 581.40 581.32
## [11] 581.44 581.68 581.17 580.53 580.01 579.91 579.14 579.16 579.55 579.67
## [21] 578.44 578.24 579.10 579.09 579.35 578.82 579.32 579.01 579.00 579.80
## [31] 579.83 579.72 579.89 580.01 579.37 578.69 578.19 578.67 579.55 578.92
## [41] 578.09 579.37 580.13 580.14 579.51 579.24 578.66 578.86 578.05 577.79
## [51] 576.75 576.75 577.82 578.64 580.58 579.48 577.38 576.90 576.94 576.24
## [61] 576.84 576.85 576.90 577.79 578.18 577.51 577.23 578.42 579.61 579.05
## [71] 579.26 579.22 579.38 579.10 577.95 578.12 579.75 580.85 580.41 579.96
## [81] 579.61 578.76 578.18 577.21 577.13 579.10 578.25 577.91 576.89 575.96
## [91] 576.80 577.68 578.38 578.52 579.74 579.31 579.89 579.96
```

```
class(huron_ts)
```

```
## [1] "ts"
```

Let's plot the data using the function `ts.plot()`. `ts.plot()` takes a times-series object, such as `huron_ts` and plots a line chart in chronological order.

```
# plots time series object (similar to line graph)
ts.plot(huron_ts, xlab = 'Year', ylab = 'Level (in feet)',
        main = 'Level of Lake Huron (1875-1972)')
```



Description of graph: As we can see, this data set doesn't prove to be extremely helpful for times series analysis (even with time as a variable), as there aren't many elements to it other than the year and a level (height) that only varies between five feet at most (it looks like a line chart, embarrassing to say at the least). So why time series analysis? Don't worry; this example was just for you to get familiar with time series objects. Thus, let's consider a different data set (something more practical) and we will spend a lot more time analyzing that data in depth.

III. Main Example: Unemployment Rate (and Inflation Rate)

Gathering the Data

For the rest of the post, we will be considering U.S. unemployment rates and inflation rates and working with this data to familiarize ourselves with real-world data. We will be gathering our data from a well-known and easily accessible website, known as FRED (Federal Reserve Economic Data), that records these monthly rates.

We will be using Civilian Unemployment Rate and CPI Annual Changes (for Inflation Rate) from the January 1948 to September 2017. All of this data is recorded monthly. We will be using the two links below to import our data (Reference 2 and Reference 3).

Reference 2: FRED Unemployment Rate - https://fred.stlouisfed.org/series/UNRATE

Reference 3: FRED Inflation Rate - https://fred.stlouisfed.org/series/CPIAUCSL#0

Note that the FRED Inflation Rate is based on the CPI index in 1982 to 1984 (just if you wanted to know). You need to specify "Change from Year Ago" to get the exact same graph.

I have downloaded the .csv files and saved them as `unrate.csv` (for unemployment rate) and `infrate.csv` (for inflation rate) in the `data` folder.

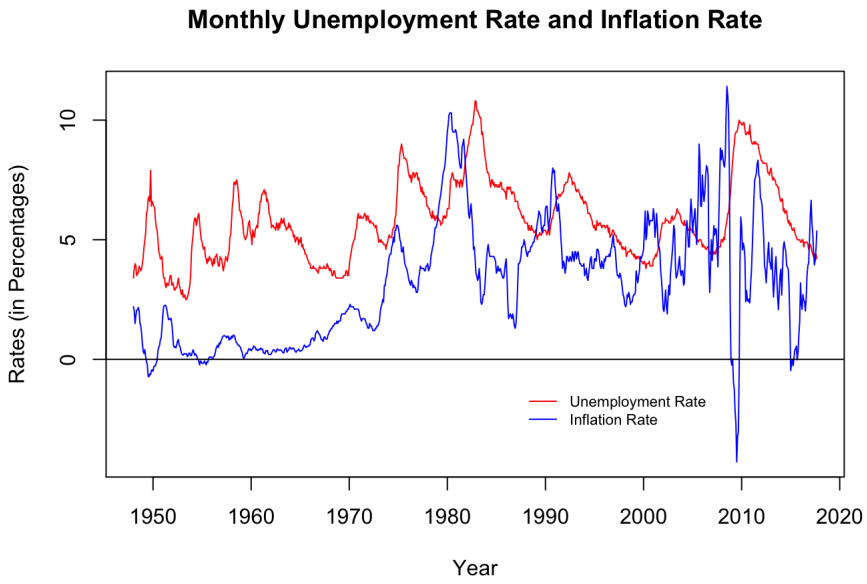
Let's create time series objects and plot these two rates by using the built-in statistics function `ts()` and `ts.plot()`.

```r
# reading the data
unrate_data <- read.csv('../data/unrate.csv')
infrate_data <- read.csv('../data/infrate.csv')

# creating ts objects (from original column names) and plotting
unrate <- ts(unrate_data$UNRATE, start = c(1948, 1), freq = 12)
infrate <- ts(infrate_data$CPIAUCSL_CH1, start = c(1948, 1), freq = 12)

# creating plots for the two variables with a legend
ts.plot(unrate, infrate, col = c('red', 'blue'),
        xlab = 'Year', ylab = 'Rates (in Percentages)',
        main = 'Monthly Unemployment Rate and Inflation Rate')

legend(1987, -1, c('Unemployment Rate', 'Inflation Rate'),
       col = c('red', 'blue'), lty = 1, bty = 'n', cex = 0.7)
abline(h = 0)
```



Description of graph: As we can see from the graph, the inflation rate seems to vary much more than the UE rate past a certain time period. **We can observe that years 2007-2009 had the biggest change in both inflation rate (sharp decline) and UE rate (sharp increase). Why? How can we quantify this relationship?** This is because of the period of economic decline, commonly known as the Great Recession. Check out the effects of the Great Recession below (Reference 4).

Reference 4: The Great Recession - https://www.federalreservehistory.org/essays/great_recession_of_200709

There seems to be an increase in UE rate as inflation rate decreases. Similarly, as UE rate decreases, inflation rate seems to increase. We can use time series analysis to gauge how unemployment rate and inflation rate are related with respect to time.

Summary: Note that it would make no sense if we took the unemployment (UE) rate from year 1950 and compared that to the inflation rate at year 2000. Time plays a factor here, so we should use it to our advantage by carefully studying its effects on UE rate and inflation rate. Let's study this graph more closely to see the relationship.

Studying the Data: Decomposition with `decompose()`

I referred to the link below to understand the decomposition of time series into its three additive categories: trend, seasonal, and random (Reference 5).

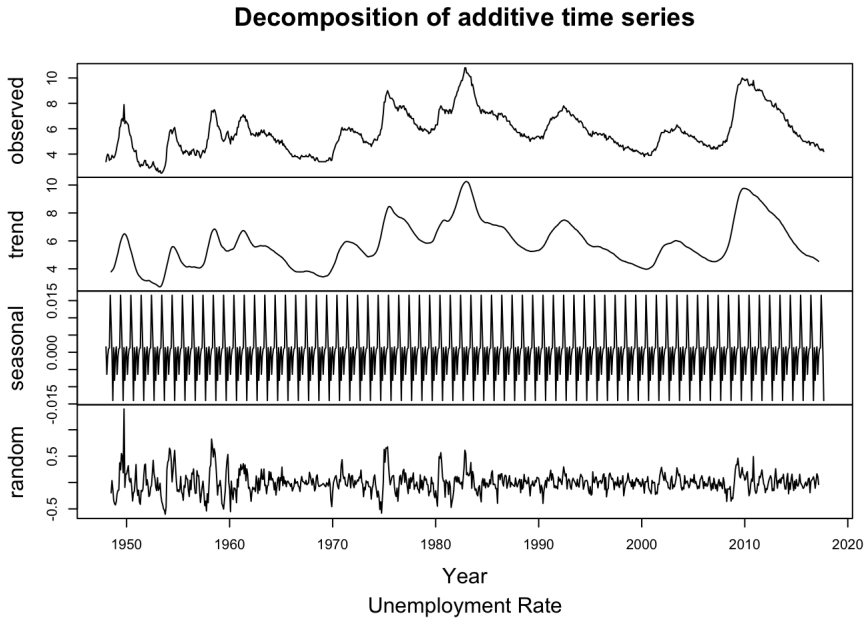
Reference 5: Decomposition of additive time series - https://anomaly.io/seasonal-trend-decomposition-in-r/

All time series are composed of the following additive components: **Observed Time Series = Trend + Seasonal + Random**

1. **Trend**: The trend decomposition allows us to see the observed time series graphed over a smooth average between each successive period. You can specify a moving averages if you wish.
2. **Seasonal**: The seasonal decomposition depicts how the seasonal patterns over a period of time have an impact on the time series. For example, UE rate increases during a certain month.
3. **Random**: This is the remainder of the "mess" after we add trend and seasonal time series components in order to achieve our original (observed) time series.

We can decompose our data into these three categories by using the function `decompose()`. If we just use `decompose()` by itself, it looks rather plain and uninteresting, so we are going to plot the data with it. Let's study the two variables, UE rate and inflation rate, separately, to see how it behaves with itself first.

```r
# plot decomposed unemployment rate data
plot(decompose(unrate), xlab = 'Year')
title(sub = list('Unemployment Rate', cex = 1))
```

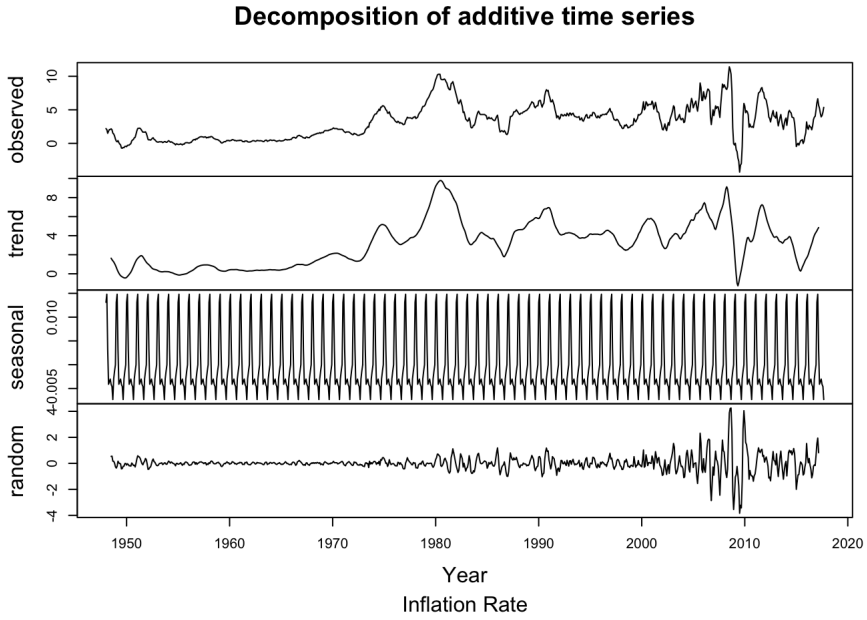


Description of graph: Four plots were graphed by default on top of each other. From these graphs, it is easy to observe how time series is composed of three main parts. There is a trend, seasonal, and random plot while decomposing the data. The observed graph is our original unemployment rate time series, whereas the trend graph is the overall pattern by smoothing the curve out so there is a moving average over the years. You can see how they look similar.

The seasonal component allows us to see how there seems to be peaks and troughs every single year. This tells us that workers seem to be laid off and then hired when needed again (and the cycle continues) – around Christmas time and summer time are the "popular" times for employment as there are large breaks. The remainder of the graph contains the "randomness" that is unexplained by either the trend and seasonal components, which depends on a variety of factors (situational variables, such as natural disaster or self-fulfilling prophecies).

Similarly, we can decompose the inflation rate time series.

```
# plot decomposed inflation rate data
plot(decompose(infrate), xlab = 'Year')
title(sub = list('Inflation Rate', cex = 1))
```



Description of graph: Again, four plots are shown above (the bottom three are the "new" ones). Observing the variability in the trend component, we can see that inflation rate can easily drop and rise from the previous year, whereas UE rate tends to go up over a large period of time and drop over successive periods), thus explaining why the trend is not as smooth. There are many endogenous variables, such as inflationary expectations, that come into play when it comes to calculating the exact reason for changes in inflation rate, so we cannot pinpoint the exact factor as to why the trend is so jagged. **Perhaps UE rate depends on the years before it? We'll study it in the next section.**

In the seasonal time series component for inflation rate, we can see its similarites with the seasonal time series component for the UE rate. This is likely correlated with the amount of breaks in the winter (holiday season) and summer, so people are more likely to spend their money on certain goods, creating variation in our data. Lastly, the randomness time series is, once again, simply the remainder of the time series graph (which can be caused by nearly any possible factor you can think of that involves some sort of spending or income).

Summary: So what can we do with these time series components? Well, there are a variety of things you are now able to study with these time series components. For instance, what can we say about relationship between UE rate and inflation rate (we saw similar seasonal trends)? Does UE rate depend on the year before it? If we know, we can make predictions for the future based on the previous years, which will allow us to identify variability between time periods and prediction of economic recessions and recoveries.

Auto-Correlation Function: `acf()`

With decomposition, we have identified a pattern with the UE rate and inflation rate. Now, we need to see if we can make predictions in the future according to the UE rate and inflation rate that we already have recorded in previous years.

We will be using the `acf()`, which stands for auto-correlation function. `acf()` takes a time series object, and its goal is to observe how the UE rate/inflation rate in a particular year is correlated to future years. For example, how do we predict the UE rate for January 2019? Will the UE rate from 1948 to 2017 tell us something about 2019's UE rate? We will answer these questions using `acf()` below.

To read more on the exact calculation on auto-correlation and what it means, I found the article lnked below quite helpful (Reference 6).

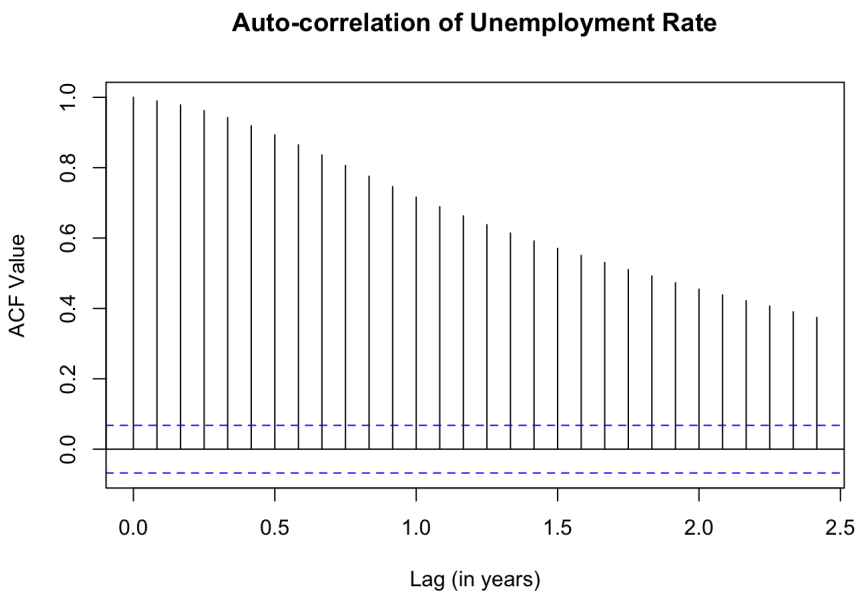
Reference 6: Autocorrelation - http://www.itl.nist.gov/div898/handbook/eda/section3/eda35c.htm

Furthermore, I used the tutorial linked below along with reading the article as it gave me a quick guide about how to use auto-correlation (Reference 7).

Reference 7: Time Series Tutorial with Correlation - https://www.youtube.com/watch?v=-ImppGbVpXI

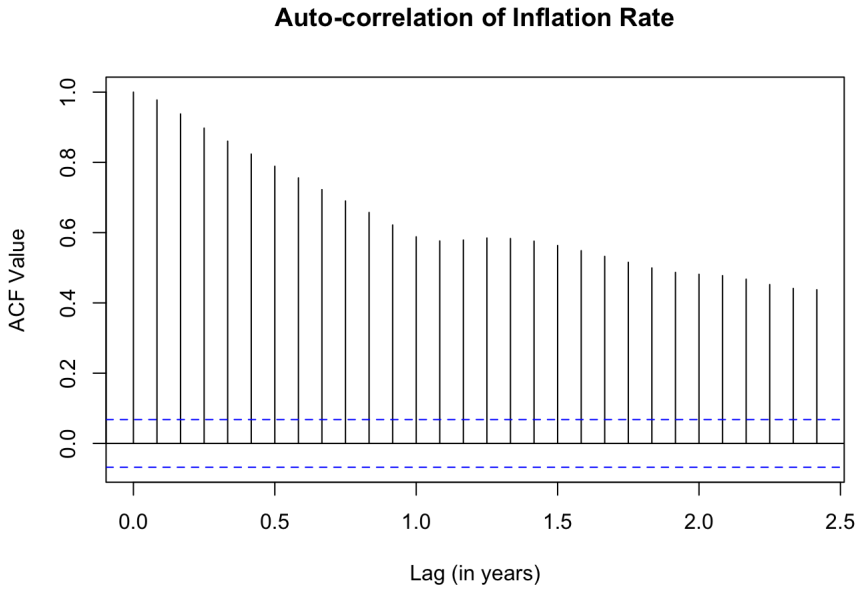
We will first see how UE rate is correlated with the months after it.

```
# using acf() on UE rate (makes a plot)
acf(unrate, main = 'Auto-correlation of Unemployment Rate',
    xlab = 'Lag (in years)', ylab = 'ACF Value')
```



Description of graph: The explanation of this graph is intuitive. We can see that the UE rate does depend on the years before it. The correlation seems to decrease over time and be less predictive of the UE rate, say, 5 years later. This makes sense and explains why periods of economic deline, such as the Great Recession, go on for an extended period of time but are sometimes predictable by statisticians and economists. We will discuss the blue dashed lines in the next section, titled "Randomness over Time."

```
# using acf() on inflation rate (makes a plot)
acf(infrate, main = 'Auto-correlation of Inflation Rate',
    xlab = 'Lag (in years)', ylab = 'ACF Value')
```



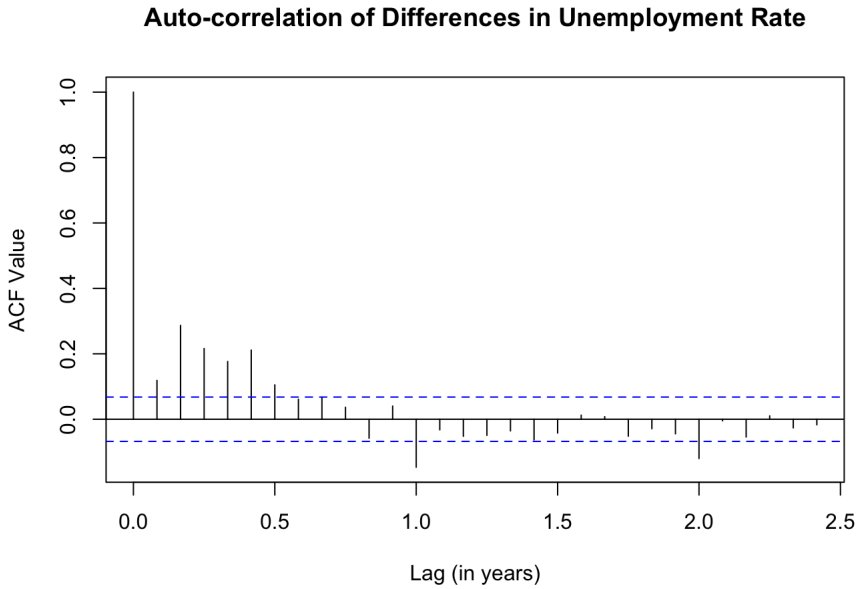
Description of graph: The explanation of this graph also makes intuitive sense, but we can see that there is actually an increase in correlation after about 1 year or so. Bumps such as these are to be studied and looked at closely. It is plausible that the increase in correlation could be because inflation during one month of the year relates to the inflation during the same month of the next year. We can see that there is quite an interesting correlation over time. Again, we will discuss the blue dashed lines in the next section, titled "Randomness over Time."

Summary: Correlation will prove to be very useful, as we can see associations between UE rate with itself in the future, as well as inflation rate with itself years later. Thus, we can forecast our data by making predictions on the future and draw graphs with regression. This is helpful for observing and studying stochastic processes, which works with the randomness category. We will be doing that in the next section.

Randomness over Time

While looking at the auto-correlation of a variable, such as UE rate, it is tempting to say that the UE rate today (2017) will not affect the UE rate three years later (2020). Let's use the `acf()` function and the `diff()` function to observe this for the unemployment rate. `diff()` calculates the difference between each time entry (month).

```
# calculates correlations of differences between times
acf(diff(unrate),
    main = 'Auto-correlation of Differences in Unemployment Rate',
    xlab = 'Lag (in years)', ylab = 'ACF Value')
```



Description of graph: Now, we can see that the auto-correlation of the differences in successive time periods in unemployment rate have negative values for large enough lag (years later) and these auto correlation values oscillate and get closer and closer to zero for large lag time. As seen but not mentioned before, the blue dotted lines represent set confidence intervals for when auto-correlation function believes that there is a possible entry that is not random. If it crosses the blue line, it means that this particular time entry in the future has a correlation that seems too unlikely to be random compared to present time.

Thus, it makes sense that this auto-correlation approaches zero, as the two entries should not depend on one another (UE rate in 2000 is not a good indicator of UE rate in 2010). An important question to ask is: **Why does it seem that 1 year and 2 year differences have a significant auto-correlation?** It is not a mistake – it actually is relevant in our data. We can see that these significant observations really correlate with each other if the entries are exactly one year apart or exactly two years apart.

Forecasting

Whether we are forecasting (predicting) the weather or the UE/inflation rate, we need to be careful and realize that we are studying each variable individually rather than together over time. This is because predictions of future UE rate allow us to generalize and predict the future inflation rate (from our observation of an indirect association between UE rate and inflation rate). Let's get started with forecasting.

We will be using the HoltWinters function to forecast our data. HoltWinters is a triple exponential smoothing method that is used for predicting multiple data points in a particular time series based on our additive seasonal components.

To read more about the exact computation for the HoltWinters model or to understand how to use the functions like I did, check out the references below. Reference 8 is a stackexchange question as to how to approach a HoltWinters approximation.

Reference 8: Forecasting With Holt-Winters Function - https://stats.stackexchange.com/questions/63250/forecasting-function-r-holt-winters-hw-approach

I also referred to the written tutorial below as it helped me understand what it means to use the HoltWinters approximation. You should check it out (Reference 9).

Reference 9: Tutorial for Holt-Winters for Dummies - https://grisha.org/blog/2016/01/29/triple-exponential-smoothing-forecasting/

Lastly, I watched multiple video tutorials to understand how to forecast, but the one I linked below is the best one that explains everything in one cohesive manner (Reference 10).

Reference 10: Video Tutorial for Times Series in R: Forecasting - https://www.youtube.com/watch?v=vypp3nXbEPE

First, we need to install the package `forecast`.

```
# need to install the 'forecast' package
install.packages("forecast")
```

Now, we will use the `HoltWinters()` function to plot the UE rate and the HoltWinters predicted UE rate. Note that there is a `HoltWinters()` that is built in, but in order to forecast, we will need to install the `forecast` package. It's not a hard function to use at all, so take a look below to see how to use it in a simple manner.

```
# creating the HoltWinters object
unrate_holt <- HoltWinters(unrate, seasonal = 'additive')
unrate_holt
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = unrate, seasonal = "additive")
##
## Smoothing parameters:
##  alpha: 0.9043078
##  beta : 0.05069537
##  gamma: 1
##
## Coefficients:
##             [,1]
## a     4.35040002
## b    -0.04512041
## s1   -0.06380772
## s2    0.01129473
## s3    0.08029558
## s4    0.11093664
## s5    0.06635179
## s6    0.01846996
## s7   -0.05505078
## s8   -0.12751368
## s9   -0.13264225
## s10 -0.16060163
## s11 -0.12102295
## s12 -0.15040002
```

The created object should be of the class, 'HoltWinters'.

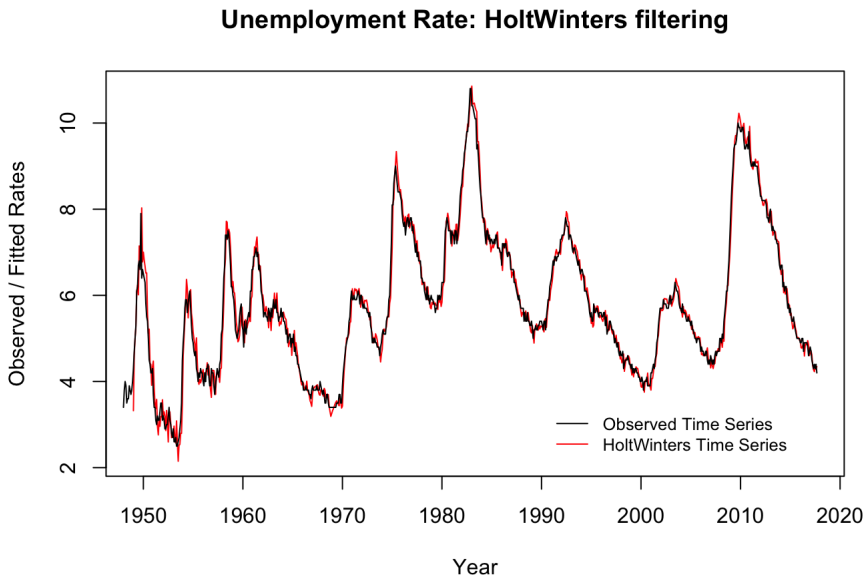
```
class(unrate_holt) # checks that it is a HoltWinters object
```

```
## [1] "HoltWinters"
```

Now, let's plot the HoltWinters data.

```
# plotting data (by default, original data in black, HoltWinters in red)
plot(unrate_holt, main = 'Unemployment Rate: HoltWinters filtering',
     xlab = 'Year', ylab = 'Observed / Fitted Rates')

legend(1990, 3.5, c('Observed Time Series', 'HoltWinters Time Series'),
       col = c('black', 'red'), lty = 1, bty = 'n', cex = 0.8)
```



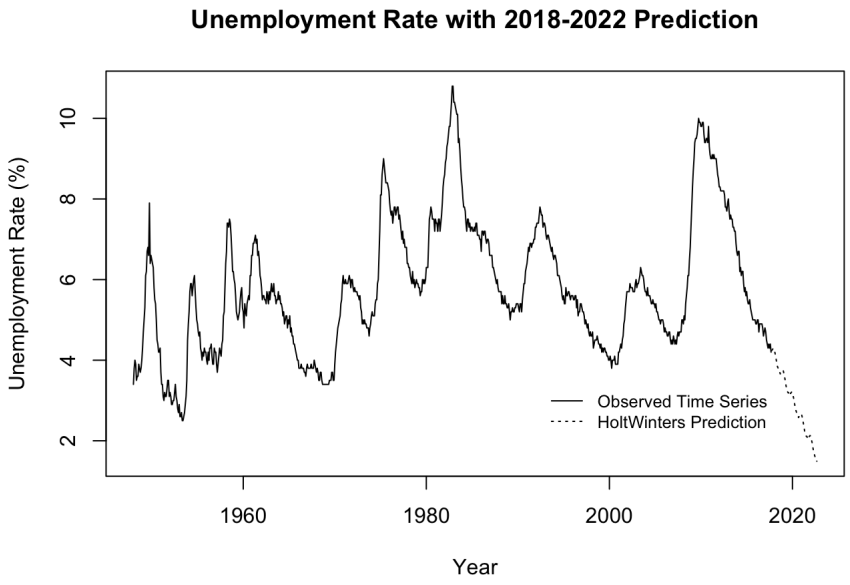
Description of graph: From this graph, we can see that the HoltWinters forecasting method generates a graph (in red) that is very similar to the original observed time series (in black). The inconsistencies come from the exponential model itself. For example, take a look at the unemployment rate from 1951 to about 1954. The HoltWinters model did not respond to the jagged observations in the original data as well as we had hoped it will. We can see that the prediction is not perfectly precise, but it does allow us to work with making predictions about the future.

Now, let's make predictions for the future using the `predict()` function and plot the predictions.

```
# predicts UE rate for next 5 years (12 * 5 months)
unrate_pred <- predict(unrate_holt, n.ahead = 12*5)

# plots both together with time
ts.plot(unrate, unrate_pred, lty = c(1,3),
        main = 'Unemployment Rate with 2018-2022 Prediction',
        xlab = 'Year', ylab = 'Unemployment Rate (%)')

legend(1992, 3.5, c('Observed Time Series', 'HoltWinters Prediction'),
               lty = c(1, 3), bty = 'n', cex = 0.8)
```



Description of graph: We can infer that the HoltWinters Prediction predicts a downward trend in UE rate and continues to do so even if we predict the next ten years. We can see this by the dotted portion of the graph above. So the main question from this graph is: *When do we stop?* This is a subjective answer, but the explanation for the downward trend is because HoltWinters uses differences in data. Since we are currently at a sharp unemployment rate decrease (economic recovery), the HoltWinters prediction will also trend downward. This downward trend results in a great downward trend in the future, but, based on past history, the prediction does not make complete sense.

So what do we do? Instead, we should avoid inappropriate extrapolation and use our intuition of past history and think about why certain predictions, such as the case above, are inaccurate. The prediction may make sense for the next year or so, but it is impossible to predict with certainty. **The main take away from this HoltWinters exercise is to see that we can model changes in future behavior, but the model is just an inconclusive measurement of a phenomenon of something that depends on so many different variables.**

Thus, we can say something about the inflation rate as well. Is it going to continue increasing? Well, let's take a look.

```
library(forecast)

# creating HoltWinters object for inflation rate
# NOTE: may see a warning/error message in R regarding optimization
infrate_holt <- HoltWinters(infrate, seasonal = "additive")
```

```
## Warning in HoltWinters(infrate, seasonal = "additive"): optimization
## difficulties: ERROR: ABNORMAL_TERMINATION_IN_LNSRCH
```

```
# predicts inflation rate for next 5 years (12 * 5 months)
infrate_pred <- predict(infrate_holt, n.ahead = 12*5)
```

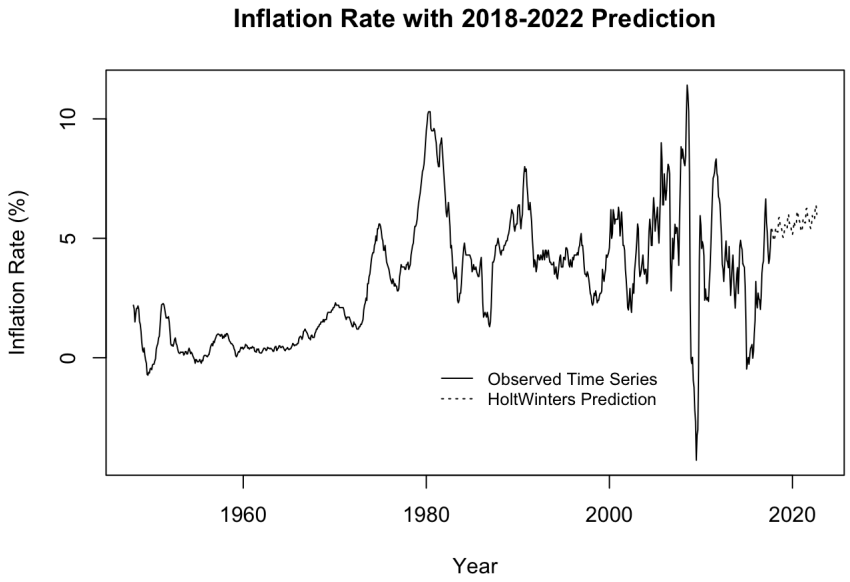
IMPORTANT NOTE: If you see a warning/error message regarding optimization on your current version of R when running the code chunk above, it is completely normal. This is because the inflation rate is so variable that when trying to smooth out the prediction, there will be a warning message. This sometimes happens when trying to use the HoltWinters package on unsmooth functions, so do not worry about it. The code chunk is still executed and names are assigned without failure. If you want to read more about it, refer below (Reference 11).

Reference 11: Abnormal Termination - https://stat.ethz.ch/pipermail/r-help/2007-February/126209.html

Now, let's plot the graph.

```
# plots both together with time
ts.plot(infrate, infrate_pred, lty = c(1,3),
        main = 'Inflation Rate with 2018-2022 Prediction',
        xlab = 'Year', ylab = 'Inflation Rate (%)')

legend(1980, 0, c('Observed Time Series', 'HoltWinters Prediction'),
               lty = c(1, 3), bty = 'n', cex = 0.8)
```



Description of graph: From this graph, we can see the an expected increase using the HoltWinters exponential prediction. This matches our observation that the unemployment rate and inflation rate are inversely related. Predictions in unemployment rate are to decrease within the next five years, whereas the inflation rate is expected to increase within the next few years. Now, the result may not be as smooth as you see above, but it should provide a general trend with additive seasonal time series.

IV. Conclusion: Take-away Message

Reference 12: Using R for Time Series Analysis - http://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/src/timeseries.html

Above (Reference 12) is an extremely useful textbook for time series if you are more interested. I used this as a guide to understanding how to use the functions that I have used above.

In our exercise, we were able to learn that unemployment rate can be predicted to a certain extent for the future. We connected this with inflation rate and studied them both separately and cohesively as one unit. This is based on the measurement of time inconsistencies and how certain variables can be predictable or unpredictable. It is a great starting point as to where we are trying to work towards – a near exact measurement. That is likely an important concept to be learned with time for linear modelling or linear/exponential regression, a class such as Statistics 151A or 153.

All in all, time series analysis is a very intricate topic to learn about and master. In our everyday lives, time affects the choices that we make and changes our “doing” to something that is now “done.” This guides us to understanding how we can use our current behaviors as records of history to our advantage in data. The objective of this post is for the reader to understand how people, as scientists, are able to understand how data can be manipulated with time as a variable. Thus, we are able to create models and apply these models with intuition, and we learn about the effects of extrapolation and how to avoid such improper assumptions in models.

V. References

All of the references are listed below in order in which they appear for ease.

Reference 1: Lake Huron - https://vincentarelbundock.github.io/Rdatasets/datasets.html

Reference 2: FRED Unemployment Rate - https://fred.stlouisfed.org/series/UNRATE

Reference 3: FRED Inflation Rate - https://fred.stlouisfed.org/series/CPIAUCSL#0

Reference 4: The Great Recession - https://www.federalreservehistory.org/essays/great_recession_of_200709

Reference 5: Decomposition of additive time series - https://anomaly.io/seasonal-trend-decomposition-in-r/

Reference 6: Autocorrelation - http://www.itl.nist.gov/div898/handbook/eda/section3/eda35c.htm

Reference 7: Time Series Tutorial with Correlation - https://www.youtube.com/watch?v=-ImppGbVpXI

Reference 8: Forecasting With Holt-Winters Function - https://stats.stackexchange.com/questions/63250/forecasting-function-r-holt-winters-hw-approach

Reference 9: Tutorial for Holt-Winters for Dummies - https://grisha.org/blog/2016/01/29/triple-exponential-smoothing-forecasting/

Reference 10: Video Tutorial for Times Series in R: Forecasting - https://www.youtube.com/watch?v=vypp3nXbEPE

Reference 11: Abnormal Termination - https://stat.ethz.ch/pipermail/r-help/2007-February/126209.html

Reference 12: Using R for Time Series Analysis - http://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/src/timeseries.html