Multivariate Modeling
DATS 6450-15
Lab # 3
Taisha Ferguson
February 12, 2020 (TF)

Abstract

The primary purpose of this lab was create an autocorrelation function and test the autocorrelation of lags in times series data. Once the autocorrelation function was created it was applied to a dataset of random values as well a time series dataset of company sales.

Introduction

In this lab I created an autocorrelation function to test the relationship between different lags in time series data. Autocorrelation is statistical function that measures the relationship between different time lags in time series data. The value of the autocorrelation can be between -1 and 1. Values closer to -1 and 1 are considered to have stronger relationships while values closer to zero are considered to have weaker relationships.

Methods, theory, and procedures

As stated in the Introduction, the autocorrelations of lags was used to examine the strength and the direction of the relationship between lags. The following formula, which was given in the assignment instructions, was used to calculate the autocorrelations:

$$\hat{\tau}_k = \frac{\sum_{t=k+1}^{T}(y_t - \overline{y})(y_{t-k} - \overline{y})}{\sum_{t=1}^{T}(y_t - \overline{y})^2}$$

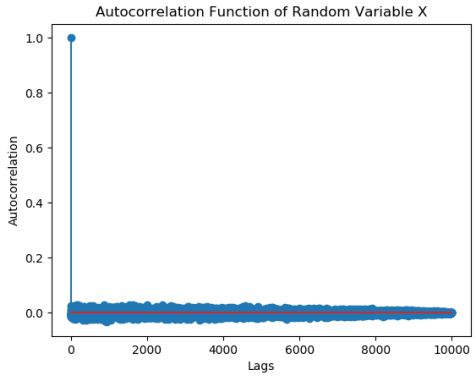
This formula was translated into Python function with the assistance of the Numpy Python Library. After the autocorrelation function was created, it was applied to the a random sample of 1000 points with mean 0 and standard deviation 1. This sample was a representation of the statistical concept of white noise. White noise means that the samples have no relationship and cannot be modeled. After running the autocorrelation function on the white noise dataset it was clear that the was in fact no relationship between lags because the values after k=0 were all very close to 0.

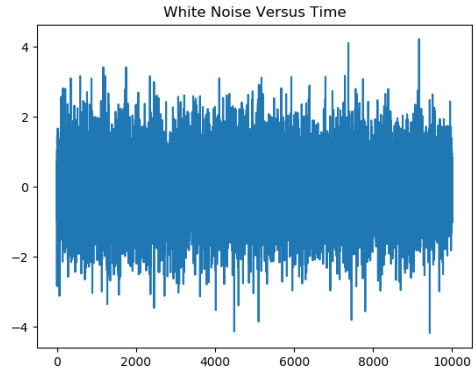
For the second portion of the assignment, the autocorrelation function was applied to a time series dataset with variables: Sales, AdBudget, and GDP. After applying the autocorrelation function and graphing the results it was clear that there was a relationship between lags for all of the variables. AdBudget had the strongest values and GDP has the lowest values.

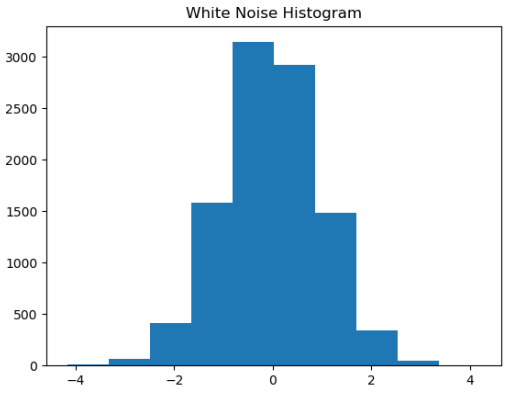
Answers to Asked Questions

4c: Write down your observations about the AFC plot, histogram and the time plot of the generated WN.

A: The ACF function of the white noise has autocorrelation values close to zero. This leads me to believe that the strength of the relationship between lags is very weak. The plot over time shows that the data is random in relationship to time. The histogram shows that the samples are normally distributed.

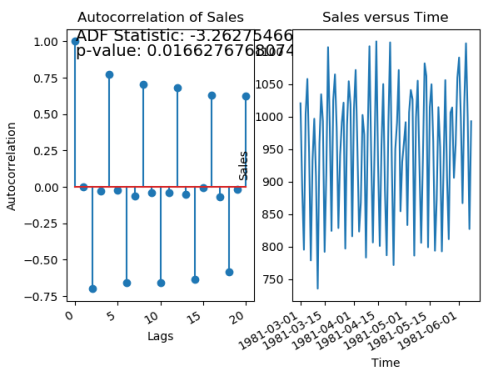


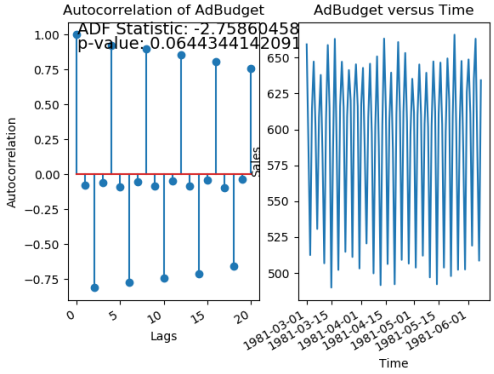


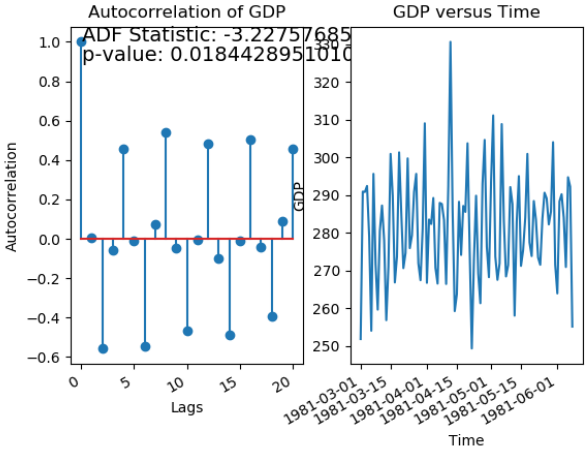


5e: Write down your observations about the correlation between stationary and non-stationary time series (if there is any) and autocorrelation function?

A: I don't see a strong correlation between the stationary and non-stationary variables except that the variable with highest autocorrelation magnitudes, AdBudget, is the only variable that is not stationary.







Conclusion

The main objective of this lab was to test the relationship of lags in times series data by calculating autocorrelation. This method was used on random variables as well as on variables from a time series data on a company's sales. There was a clear difference in results between the random variables and the company sales data. This difference supports the conclusion that there was no relationship between the lags of the random variables and there were varying degrees of relationship between the different company sales variables

Appendix A – Handwritten Computations

(2) $Y_{(t)} = [3, 9, 27, 81, 243]$

$$T_k = \frac{\sum_{t=k+1}^{T} (Y_t - \bar{Y})(Y_{t-k} - \bar{Y})}{\sum_{t=1}^{T} (Y_t - \bar{Y})^2}$$

$$\bar{Y} = \frac{3+9+27+81+243}{5} = 72.6$$

T_k (denominator) $= \sum_{t=1}^{T} (Y_t - \bar{Y})^2$

$= (3-72.6)^2 + (9-72.6)^2 + (27-72.6)^2 + (81-72.6)^2 + (243-72.6)^2$

$= 4844.16 + 4044.96 + 70.56 + 2079.36 + 29036.16$

$= 40075.2$

T_0 (numerator) $= \sum_{t=0+1}^{5} (Y_1 - \bar{Y})(Y_1 - \bar{Y})$

$= (3-72.6)^2 + (9-72.6)^2 + (27-72.6)^2 + (81-72.6)^2 + (243-72.6)^2$

$= 40075.2$

$$\hat{T}_0 = \frac{40075.2}{40075.2} = 1$$

T_1 (numerator) $= \sum_{t=1+1}^{5} (Y_2 - \bar{Y})(Y_1 - \bar{Y})$

$= (9 - 72.6)(3 - 72.6) + (27 - 72.6)(9 - 72.6) + (81 - 72.6)(27 - 72.6)$
$+ (243 - 72.6)(81 - 72.6)$

$= 4426.56 + 2878.536 + -380.184 + 1431.36$

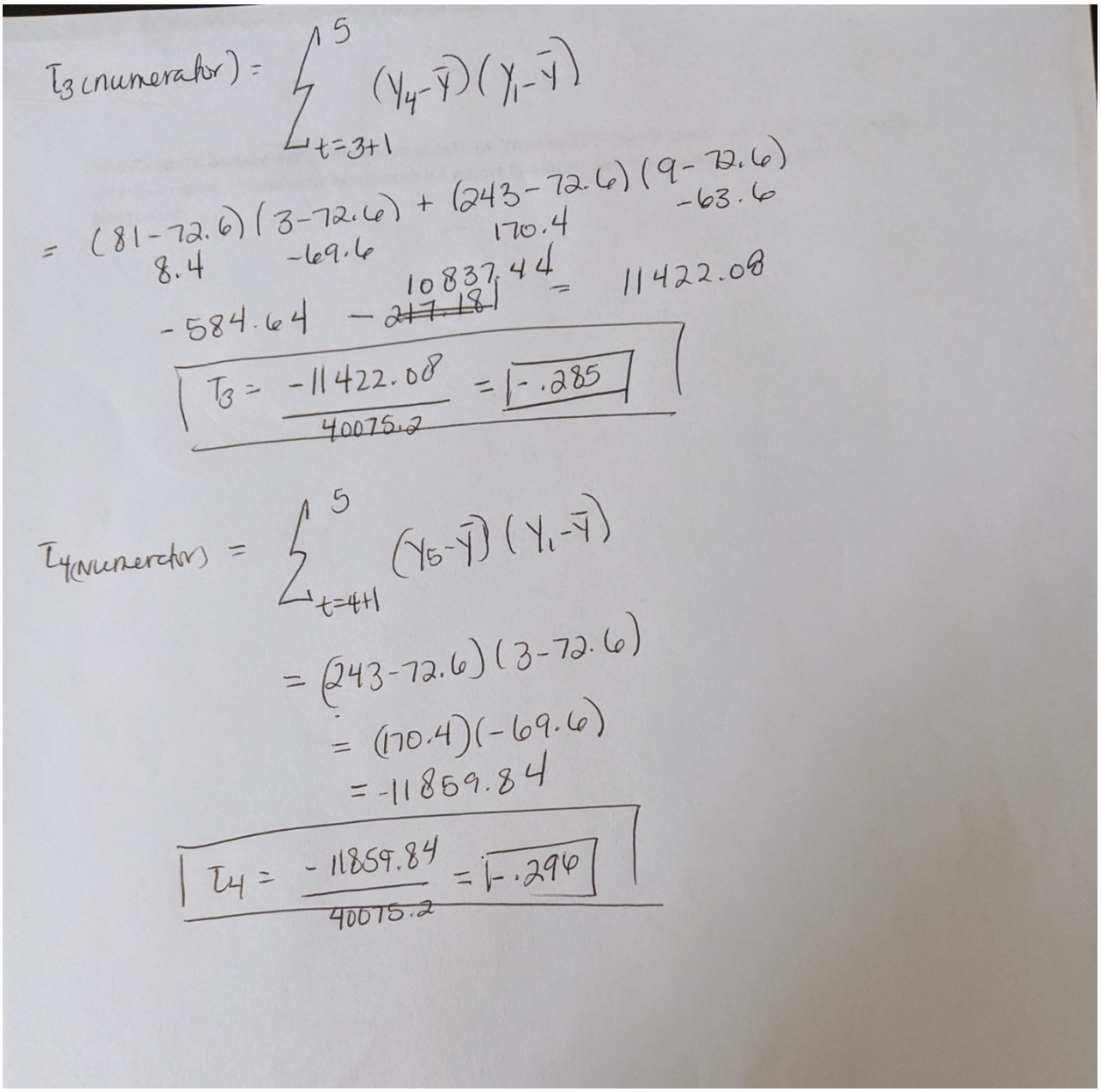
$r_1 = \frac{8356.272}{40075.2} = 0.21$

T_2 (numerator) $= \sum_{t=2+1}^{5} (Y_3 - \bar{Y})(Y_1 - \bar{Y})$

$= (27 - 72.6)(3 - 72.6) + (81 - 72.6)(9 - 72.6) + (243 - 72.6)(27 - 72.6)$

$3150.096 + -534.24 - 7694.20$

$r_2 = \frac{-5078.34}{40075.2} = -.127$



Appendix B – Python Code

```
# Import Packages
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from statsmodels.tsa.stattools import adfuller

# 3 - Create white noise with zero mean and standard deviation of 1 and 10000 samples
X = 1*np.random.randn(10000) + 0
```

```python
# 4 - Write Python Code to estimate Autocorrelation Function
def autocorrelation(y):
    y_bar = np.sum(y) / len(y)
    acf_list = []
    numerator = 0
    denomintator = 0
    n = len(y)
    for i in range(len(y)):
        denomintator += np.square(y[i] - y_bar)
    p = 0
    while p < n:
        r = n - p
        for j, z in zip(range(p, n), range(0, r)):
            numerator += ((y[j] - y_bar) * (y[z] - y_bar))
        quotient = numerator / denomintator
        acf_list.append(quotient)
        numerator = 0
        p = p + 1
    return acf_list


# 4a - Plot the ACF for the generated data in step 3.
#The ACF needs to be plotted using “stem” command.
AcfX= autocorrelation(X)
plt.stem(AcfX)
plt.title("Autocorrelation Function of Random Variable X")
plt.xlabel("Lags")
plt.ylabel("Autocorrelation")
plt.show()


# 4b - Plot both the generated WN in step 3 versus time andplot the histogram.

plt.plot(X)
plt.title("White Noise Versus Time")
plt.show()
plt.hist(X)
plt.title("White Noise Histogram")
plt.show()

# # 5 - Load the time series dataset tute1.csv (fromLAB#1)

df=pd.read_csv("tute1.csv", index_col="Date", parse_dates=True)
df=df[['Sales', 'AdBudget', 'GDP']]
print(df.head())

# 5a - Using python code written in the previous step,
# plot the ACF for the “Sales” and “Sales” versus time next to each other.
# You can use subplot command.

AcfSales = autocorrelation(df["Sales"])
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.stem(AcfSales)
ax2.plot(df['Sales'])
ax1.set_title("Autocorrelation of Sales")
ax1.set_ylabel('Autocorrelation')
ax1.set_xlabel('Lags')
ax2.set_title("Sales versus Time")
ax2.set_xlabel('Time')
ax2.set_ylabel('Sales')
fig.autofmt_xdate()
```

```
plt.show()


# 5b - Using python code written in the previous step,
# plot the ACF forthe “AdBudget” and “AdBudegt” versus time next to each other. You
can use subplot command.
AcfAdBudget = autocorrelation(df["AdBudget"])
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.stem(AcfAdBudget)
ax2.plot(df['AdBudget'])
ax1.set_title("Autocorrelation of AdBudget")
ax1.set_ylabel('Autocorrelation')
ax1.set_xlabel('Lags')
ax2.set_title("AdBudget versus Time")
ax2.set_xlabel('Time')
ax2.set_ylabel('AdBudget')
fig.autofmt_xdate()
plt.show()

# # 5c - Using python code written in the previous step,
# # plot the ACF for the “GDP” and “GDP” versus time next to each other. You can use
subplot command.
AcfGDP = autocorrelation(df["GDP"])
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.stem(AcfGDP)
ax2.plot(df['GDP'])
ax1.set_title("Autocorrelation of GDP")
ax1.set_ylabel('Autocorrelation')
ax1.set_xlabel('Lags')
ax2.set_title("GDP versus Time")
ax2.set_xlabel('Time')
ax2.set_ylabel('GDP')
fig.autofmt_xdate()
plt.show()

# # 5d - Run the ADF-testfor part a , b, and c and display them next to ACF and time
plot in each section.
def ADF_Cal(x):
    result = adfuller(x)
    print("ADF Statistic: %f" %result[0])
    print("p-value: %f" % result[1])
    print("Critical Values:")
    a= print("ADF Statistic: %f" %result[0])
    b= print("p-value: %f" % result[1])
    for key, value in result[4].items():
        print('\t%s: %.3f' %(key, value))
    return result

result=ADF_Cal(df['Sales'])
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.stem(AcfSales[:21])
ax2.plot(df['Sales'])
ax1.set_title("Autocorrelation of Sales")
ax1.set_ylabel('Autocorrelation')
ax1.set_xlabel('Lags')
ax2.set_title("Sales versus Time")
ax2.set_xlabel('Time')
ax2.set_ylabel('Sales')
fig.autofmt_xdate()
text1 = "ADF Statistic: " + str(result[0])
text2 = "p-value: " + str(result[1])
```

```
ax1.text(.1,1, text1, fontsize=14)
ax1.text(.1,.9, text2, fontsize=14)
plt.show()

result=ADF_Cal(df['AdBudget'])
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.stem(AcfAdBudget[:21])
ax2.plot(df['AdBudget'])
ax1.set_title("Autocorrelation of AdBudget")
ax1.set_ylabel('Autocorrelation')
ax1.set_xlabel('Lags')
ax2.set_title("AdBudget versus Time")
ax2.set_xlabel('Time')
ax2.set_ylabel('Sales')
fig.autofmt_xdate()
text1 = "ADF Statistic: " + str(result[0])
text2 = "p-value: " + str(result[1])
ax1.text(.1,1, text1, fontsize=14)
ax1.text(.1,.9, text2, fontsize=14)
plt.show()


result=ADF_Cal(df['GDP'])
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.stem(AcfGDP[:21])
ax2.plot(df['GDP'])
ax1.set_title("Autocorrelation of GDP")
ax1.set_ylabel('Autocorrelation')
ax1.set_xlabel('Lags')
ax2.set_title("GDP versus Time")
ax2.set_xlabel('Time')
ax2.set_ylabel('GDP')
fig.autofmt_xdate()
text1 = "ADF Statistic: " + str(result[0])
text2 = "p-value: " + str(result[1])
ax1.text(.1,1, text1, fontsize=14)
ax1.text(.1,.9, text2, fontsize=14)
plt.show()

# # 5f - The number lags used for this question is 20.
```