# **Jacob Wren, Cal Poly, San Luis Obispo, CA**

# **Megan Dibble, Cal Poly, San Luis Obispo, CA**

# **Introduction**

With the wide availability of time series data, this paper seeks to investigate the SAS procedure PROC TIMESERIES to share insights on the procedure’s purpose and usage.

**Time Series Data**

Data is referred to as time series data if the observations are repeatedly collected at uniformly spaced intervals.[[1]](#footnote-0) An important differentiating characteristic of time series data is that the order of the data points matter.[[2]](#footnote-1) Time series data can be collected in a wide variety of situations, from production operations to stock prices that are measured over time.

**PROC TIMESERIES**

When analyzing time series data, a major goal is to analyze the underlying structure of the data. PROC TIMESERIES can perform many of these analyses including, but not limited to:

* seasonal decomposition/adjustment to estimate (and potentially remove) seasonal effects
* correlation analysis to look at autocovariance, autocorrelation, partial autocorrelation, inverse autocorrelation, and white noise statistics
* spectral analysis to identify dominant cycles (utilizing periodograms)

PROC TIMESERIES, with the use of SAS ODS Graphics and the PROC SGPLOT procedure, can also create a wide array of explanatory graphs. Analyzing time series data can be motivated by the goal of predicting future observations, so this procedure fits nicely with PROC FORECAST as well.

After an overview of the syntax and and options of PROC TIMESERIES, analysis of financial time series datasets will be performed, highlighting some of the applications of the procedure.

**Syntax**

PROC TIMESERIES DATA=<input-data-set>  
 OUT=<output-data-set>;  
 ID <time-ID-variable> INTERVAL=<frequency>  
 ACCUMULATE=<statistic>;  
 VAR <time-series-variables>;  
 RUN;

The resulting time series data is stored in the data set <output-data-set>. The INTERVAL= <frequency> option specifies that the transactions are to be accumulated on a <frequency> basis. The ACCUMULATE=<statistic> option specifies that the <statistic> is to be calculated by frequency. After the transactional data is accumulated into a time series format, many of the procedures provided with SAS/ETS software can be used to analyze the resulting time series data.

**General Syntax with Options**

PROC TIMESERIES Statement;

proc timeseries options;

BY variables ;

A BY statement can be used with PROC TIMESERIES to obtain separate dummy variable definitions for groups of observations defined by the BY variables.

CORR statistics-list / options ;

A CORR statement can be used with the TIMESERIES procedure to specify options related to time domain analysis of the accumulated time series. Only one CORR statement is allowed.

CROSSCORR statistics-list / options ;

A CROSSCORR statement can be used with the TIMESERIES procedure to specify options that are related to cross-correlation analysis of the accumulated time series. Only one CROSSCORR statement is allowed.

CROSSVAR/VAR variable-list / options ;

The VAR and CROSSVAR statements list the numeric variables in the DATA= data set whose values are to be accumulated to form the time series.

DECOMP component-list / options ;

A DECOMP statement can be used with the TIMESERIES procedure to specify options related to classical seasonal decomposition of the time series data. Only one DECOMP statement is allowed.

ID variable INTERVAL= interval-option ;

The ID statement names a numeric variable that identifies observations in the input and output data sets. The ID variable’s values are assumed to be SAS date or datetime values. In addition, the ID statement specifies the (desired) frequency associated with the time series

SEASON statistics-list / options ;

A SEASON statement can be used with the TIMESERIES procedure to specify options that are related to seasonal analysis of the time-stamped transactional data. Only one SEASON statement is allowed.

SPECTRA statistics-list / options ;

A SPECTRA statement can be used with the TIMESERIES procedure to specify which statistics appear in the OUTSPECTRA= data set. The SPECTRA statement options are used in performing a spectral analysis on the variables listed in the VAR statement.

SSA / options ;

An SSA statement can be used with the TIMESERIES procedure to specify options that are related to singular spectrum analysis (SSA) of the accumulated time series. Only one SSA statement is allowed.

TREND statistics-list / options ;

A TREND statement can be used with the TIMESERIES procedure to specify options related to trend analysis of the time-stamped transactional data. Only one TREND statement is allowed.

|  |
| --- |
| **Required Statements**  PROC TIMESERIES DATA=<datasetname>  OUT=<where data is sent>  \*Note this is where optional statements go\*  ID <varname> INTERVAL=<interval>  ACCUMULATE=<option>;  VAR <varname>;  RUN;  **Optional Statements**  **CROSSPLOTS= *option* | ( *options* )**  specifies the cross-variable graphical output desired. By default, the TIMESERIES procedure produces no graphical output. The following plotting options are available:   * SERIES: plots the time series (OUT= data set). * CCF: plots the cross-correlation functions (OUTCROSSCORR= data set). * ALL: same as PLOTS=(SERIES CCF).   For example, CROSSPLOTS=SERIES plots the two time series. The CROSSPLOTS= option produces graphical output for these results by using the Output Delivery System (ODS). The CROSSPLOTS= option produces results similar to the data sets listed in parentheses next to the preceding options.  **MAXERROR= *number***  limits the number of warning and error messages that are produced during the execution of the procedure to the specified value. The default is MAXERRORS=50. This option is particularly useful in BY-group processing where it can be used to suppress the recurring messages.  **OUT= *SAS-data-set***  names the output data set to contain the time series variables specified in the subsequent VAR and CROSSVAR statements. If BY variables are specified, they are also included in the OUT= data set. If an ID variable is specified, it is also included in the OUT= data set. The values are accumulated based on the ID statement INTERVAL= or the ACCUMULATE= option or both. The OUT= data set is particularly useful when you want to further analyze, model, or forecast the resulting time series with other SAS/ETS procedures.  **OUTCORR= *SAS-data-set***  names the output data set to contain the univariate time domain statistics.  **OUTCROSSCORR= *SAS-data-set***  names the output data set to contain the cross-correlation statistics.  **OUTDECOMP= *SAS-data-set***  names the output data set to contain the decomposed and/or seasonally adjusted time series.  **OUTPROCINFO= *SAS-data-set***  names the output data set to contain information in the SAS log, specifically the number of notes, errors, and warnings and the number of series processed, analyses requested, and analyses failed.  **OUTSEASON= *SAS-data-set***  names the output data set to contain the seasonal statistics. The statistics are computed for each season as specified by the ID statement INTERVAL= option or the PROC TIMESERIES statement SEASONALITY= option. The OUTSEASON= data set is particularly useful when analyzing transactional data for seasonal variations.  **OUTSPECTRA= *SAS-data-set***  names the output data set to contain the univariate frequency domain analysis results.  **OUTSSA= *SAS-data-set***  names the output data set to contain the singular spectrum analysis result series.  **OUTSUM= *SAS-data-set***  names the output data set to contain the descriptive statistics. The descriptive statistics are based on the accumulated time series when the ACCUMULATE= and/or SETMISSING= options are specified in the ID or VAR statements. The OUTSUM= data set is particularly useful when analyzing large numbers of series and a summary of the results are needed.  **OUTTREND= *SAS-data-set***  names the output data set to contain the trend statistics. The statistics are computed for each time period as specified by the ID statement INTERVAL= option. The OUTTREND= data set is particularly useful when analyzing transactional data for trends.  **PLOTS= *option* | ( *options* )**  specifies the univariate graphical output desired. By default, the TIMESERIES procedure produces no graphical output. The following plotting options are available:   * SERIES: plots the time series (OUT= data set). * RESIDUAL: plots the residual time series (OUT= data set). * HISTOGRAM: plots a histogram of the time series values * CYCLES: plots the seasonal cycles (OUT= data set). * CORR: plots the correlation panel (OUTCORR= data set). * ACF: plots the autocorrelation function (OUTCORR= data set). * PACF: plots the partial autocorrelation function (OUTCORR= data set). * IACF: plots the inverse autocorrelation function (OUTCORR= data set). * WN: plots the white noise probabilities (OUTCORR= data set). * DECOMP: plots the seasonal adjustment panel (OUTDECOMP= data set). * TCS: plots the trend-cycle-seasonal component (OUTDECOMP= data set). * TCC: plots the trend-cycle component (OUTDECOMP= data set). * SIC: plots the seasonal-irregular component (OUTDECOMP= data set). * SC: plots the seasonal component (OUTDECOMP= data set). * SA: plots the seasonal adjusted component (OUTDECOMP= data set). * PCSA: plots the percent change in the seasonal adjusted component (OUTDECOMP= data set). * IC: plots the irregular component (OUTDECOMP= data set). * TC: plots the trend component (OUTDECOMP= data set). * CC: plots the cycle component (OUTDECOMP= data set). * PERIODOGRAM<(option)>: plots the periodogram (OUTSPECTRA= data set). The available options for modifying the periodogram are:   + MAXFREQ=number: specifies the maximum frequency in radians to include in the plot   + MINPERIOD=number: specifies the minimum period to include in the plot * SPECTRUM<(option)>: plots the spectral density estimate (OUTSPECTRA= data set). The available options for modifying the spectrum plot are:   + MAXFREQ=number: specifies the maximum frequency in radians to include in the plot   + MINPERIOD=number: specifies the minimum period to include in the plot * SSA: plots the singular spectrum analysis results (OUTSSA= data set). * ALL: same as PLOTS=(SERIES HISTOGRAM ACF PACF IACF WN SSA PERIODOGRAM SPECTRUM). * BASIC: same as PLOTS=(SERIES HISTOGRAM CYCLES CORR DECOMP) * PRINT= option | ( options ): specifies the printed output desired. By default, the TIMESERIES procedure produces no printed output. The following printing options are available:   + TRENDS: prints the trend statistics table (OUTTREND= data set)   + DECOMP: prints the seasonal decomposition/adjustment table (OUTDECOMP= data set).   + SEASONS: prints the seasonal statistics table (OUTSEASON= data set).   + DESCSTATS: prints the descriptive statistics for the accumulated time series (OUTSUM= data set).   + SUMMARY: prints the descriptive statistics table for all time series (OUTSUM= data set).   + SSA: prints the singular spectrum analysis results (OUTSSA= data set).   + ALL: same as PRINT=(DESCSTATS SUMMARY).   **PRINTDETAILS**  specifies that output requested with the PRINT= option be printed in greater detail.  **SEASONALITY= *number***  specifies the length of the seasonal cycle. For example, SEASONALITY=3 means that every group of three time periods forms a seasonal cycle. By default, the length of the seasonal cycle is one (no seasonality) or the length implied by the INTERVAL= option specified in the ID statement. For example, INTERVAL=MONTH implies that the length of the seasonal cycle is 12.  **SORTNAMES**  specifies that the variables specified in the VAR and CROSSVAR statements be processed in sorted order by the variable names. This option allows the output data sets to be presorted by the variable names.[[3]](#footnote-2) |
|  |

# 

# **Examples**

PROC TIMESERIES has a host of applications, and this paper focuses on the general usefulness of the procedure to analyze and visualize financial data.

## **Example 1: Plotting Stock Price Data**

This dataset contains data on all of the S&P 500 company stocks for the past five years ending in February of 2018. It was downloaded from the website Kaggle.com and created using The Investor's Exchange API. The data were collected daily, and variables include the date, the open, high, low and closing price per share, the number of shares traded, and the stock’s ticker name.

For the first analysis, the data was sorted to analyze only Apple stock (AAPL). The code below was used to generate some initial impressions of the data[[4]](#footnote-3).

ods graphics on;

proc timeseries data = stocks2 print = descstats plots = (series corr);

var high;

id date2 interval=day;

label date2=date;

run;

ods graphics off;

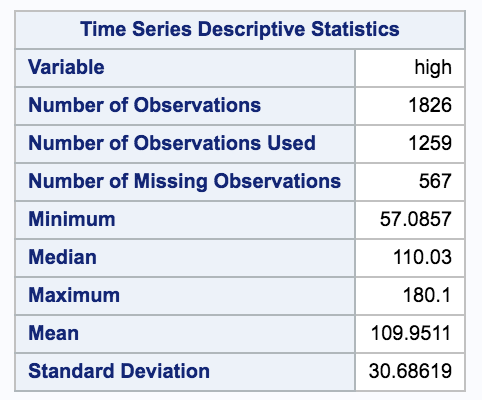
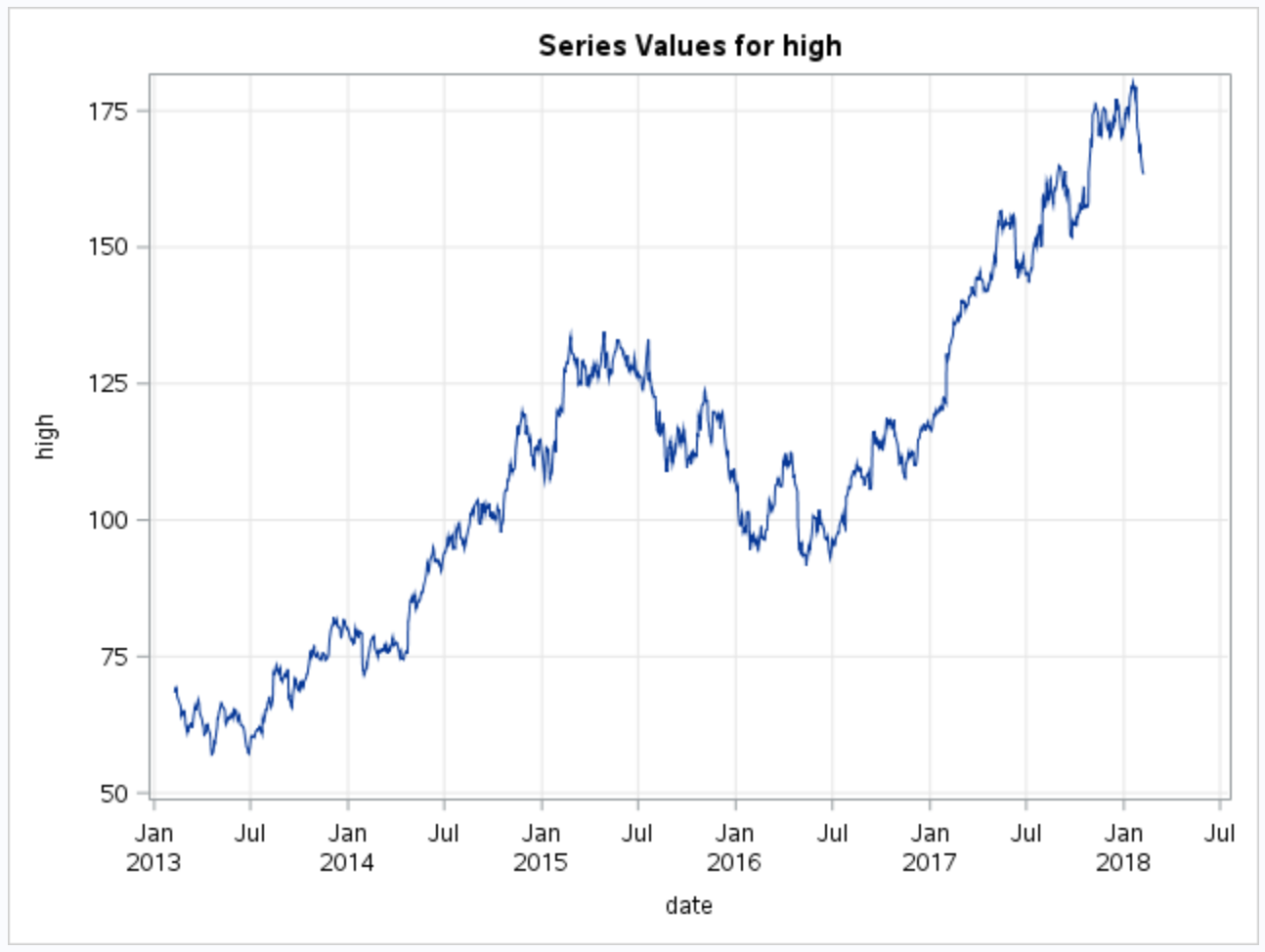
This code produced the table and plots below:

Table 1: SAS Descriptive Statistics

Figure 1: Outputted Series Plot

## 

## 

## 

## 

## 

Figure 2: Outputted Correlation Panel

Table 1 contains descriptive statistics for the variable selected in the var statement. The PROC TIMESERIES used above also outputs a series plot (Figure 1) so the user can view any obvious trends or patterns for the selected variable over the entire timeline of the data. The PLOTS = CORR option resulted in Figure 2, which plots the autocorrelation function (top left), the partial autocorrelation function (top right), the inverse autocorrelation function (bottom left) and the white noise probabilities, all of which can be used to investigate autocorrelation relationships as the lag between points increases.

The following code was ran to display the trends in the low prices over the last 5 years for Apple stock. Notice the PROC TIMESERIES outputted the trends to a temporary dataset titled trend, where the interval was quarters, and then PROC SGPLOT was utilized to visualize the trends[[5]](#footnote-4).

ods graphics on;

proc timeseries data = stocks2

out=series

outtrend=trend;

var low;

id date2 interval=qtr accumulate=avg;

run;

proc print data = trend (obs=4);

title "Trend Statistics";

run;

title1 "Trend Statistics Graph";

proc sgplot data=trend;

series x=date2 y=max / lineattrs=(pattern=solid);

series x=date2 y=mean / lineattrs=(pattern=solid);

series x=date2 y=min / lineattrs=(pattern=solid);

yaxis display=(nolabel);

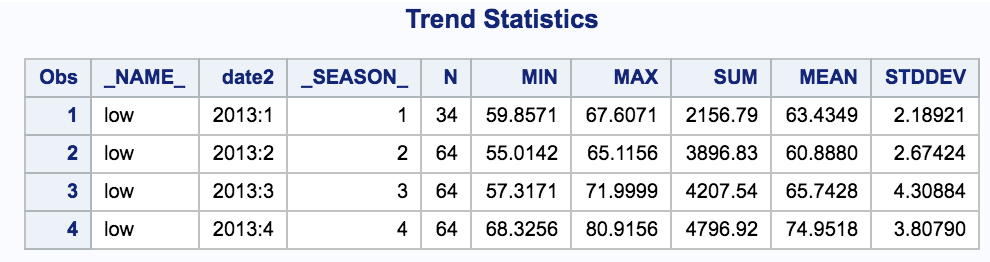
format date2 year4.;

label date2 = date;

run;

ods graphics off;

The PROC PRINT (see Figure 3) was used to illustrate the statistics that are included in the generated trend data (note only 4 observations are shown). Figure 4 displays the trend behavior over time for the high price of Apple, along with minimum and maximum values to gauge the true width of the trend data.

Figure 3: Trend Statistics

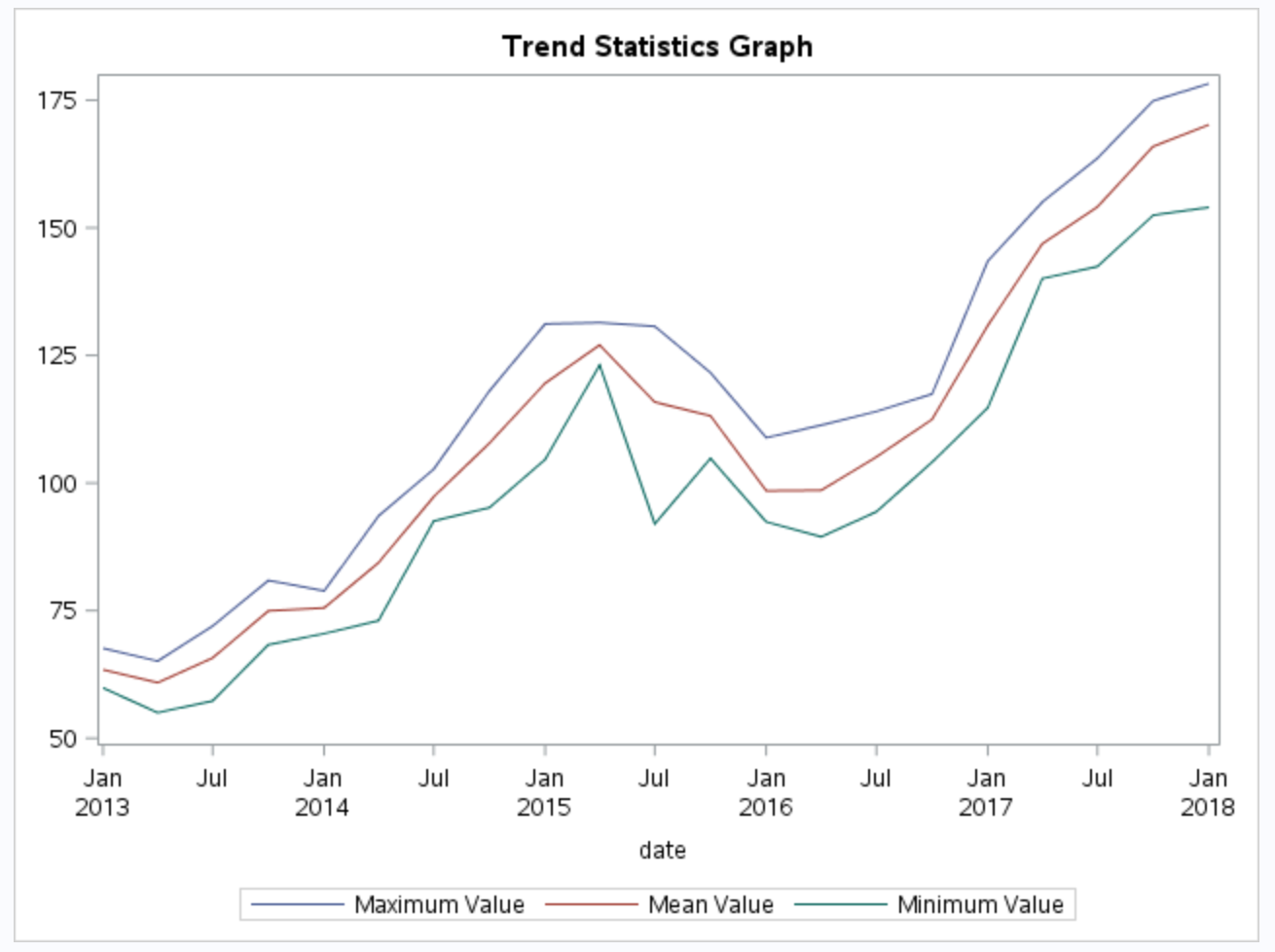


Figure 4: Trend Statistics graph

To examine if there was cyclical behavior in the high prices of **all** S&P 500 stocks over time, the periodogram option can be utilized. In this example, the data for the high price of all stocks on the same day was averaged in a data step to look at cumulative trends. Then PROC TIMESERIES was utilized[[6]](#footnote-5) :

proc sort data = stocks;

by date2;

run;

data allstocks;

set stocks;

by date2;

retain hightot;

if first.date2 = 1 then do;

obs = 0;

avg = 0;

hightot = 0;

end;

obs + 1;

hightot = hightot + high;

avg = hightot / obs;

if last.date2 = 1 then output;

run;

proc print data = allstocks (obs=5);

run;

proc timeseries data = allstocks plots = (periodogram);

var avg;

id date2 interval=day;

label date2=date;

run;

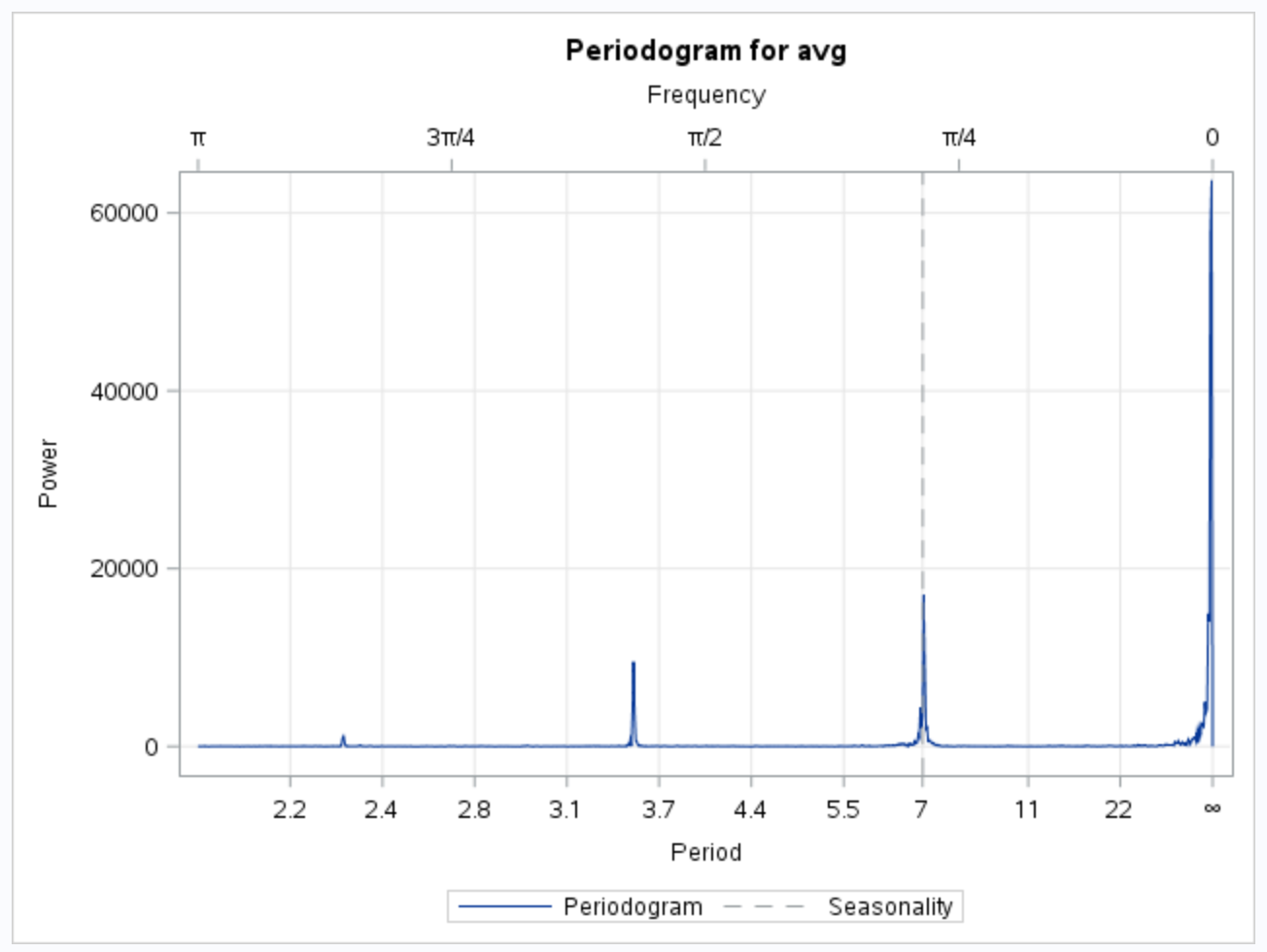


Figure 5: Periodogram for average high price of S&P 500 stocks

The figure above depicts the cyclical behavior of the market. The seasonality finding of a period of seven days makes intuitive sense as the interval is day and there are seven days in the week.

## 

## 

## **Example 2: Plotting Bitcoin data**

This dataset contains daily data from the last five years on the several key variables of bitcoin. The dataset is timestamped and has the four different value of the price: The daily open, high, low, and close. It also has the volume of daily trades and market capitalization. All values are in US dollars.

Imported bitcoindata.Csv and created a data set called bit. The first 100 rows were checked to determine the appropriate type and length for variables.

proc import

datafile= "/folders/myfolders/SAS-330/Bitcoindata.csv"

out = bit

dbms = csv

replace;

guessingrows=100;

run;

Printed first five observations from bit data set.

proc print data = bit (obs=5);

run;

Table 2: SAS data set



The variables close and market\_cap need to be changed to numeric. Because they have commas proc import reads them in as character. In order to use them in proc timeseries they must be numeric.

A data step was used along with the input function to convert them to numeric variables. This created two new variables close1 and market\_cap1.

data bit1;

set bit;

close1=INPUT(close,comma9.);

market\_cap1=INPUT(Market\_Cap,comma15.);

Run;

First five observations from data set bit1 are printed. Notice close1 and market\_cap1 are included.

proc print data = bit1 (obs=5);

run;

Table 3: SAS data set with new variables



For proc timeseries the id variable must be sorted. The id variable is date in this case.

It is also important to note that data was originally sorted in ascending order thus proc timeseries did not like this and date had to be resorted. By default it was sorted in descending order.

proc sort data = bit1 out = bit2;

by date;

run;

Data set bit3 was created to work with sorted data from bit2.

data bit3;

set bit2;

run;

proc print data = bit3(obs=5);

run;

Table 4: SAS data sorted



Proc timeseries used for close1 variable. Bitcoin is the output data set. I have asked for a summary of the data and trend and seasonal analysis as well as three plots (series correlation and seasonal decomposition).

proc timeseries data=bit3 plot=(series corr decomp)

out=bitcoin

outsum=bit7 print = summary

outtrend=trend print = trends

outseason=bit8 print = seasons;

id date interval=month

accumulate=total;

var close1;

run;

Table 5: SAS trend statistics

Have statistics for sixty months (five years of data). Only 19 showed.

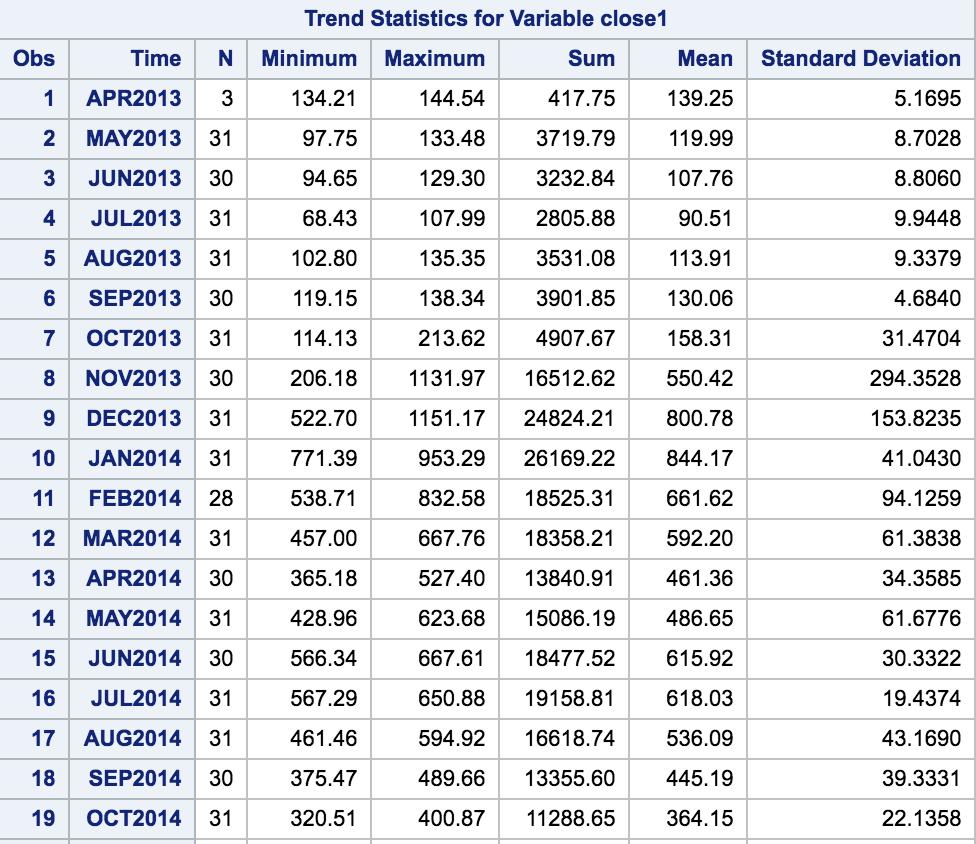


Table 6: Season statistics

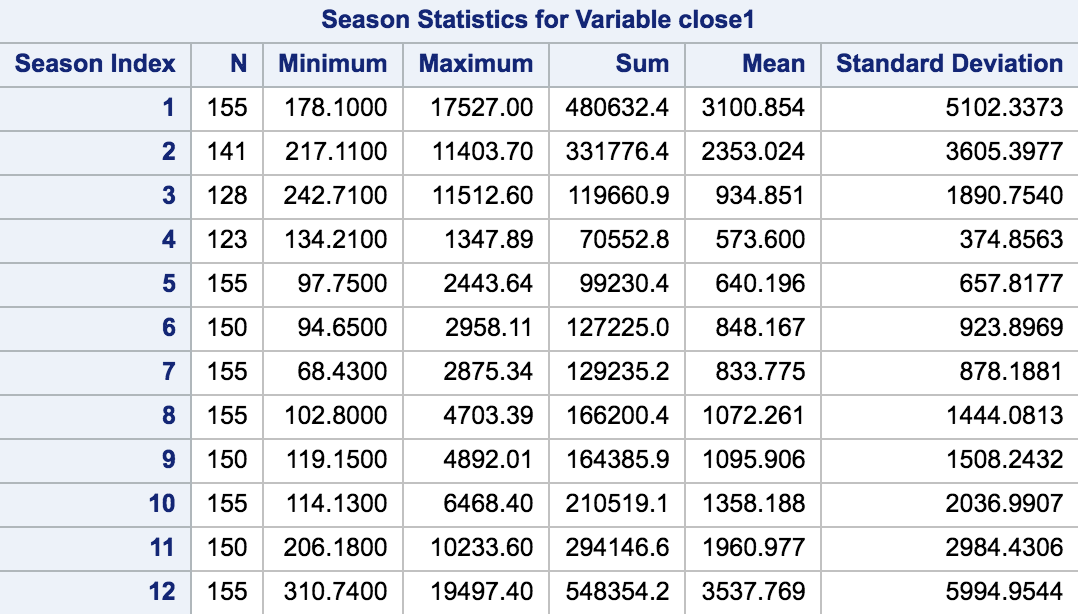
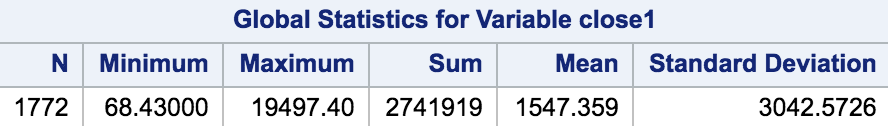


Table 7: SAS global statistics



\*Note bitcoin did pass 20,000 but that was a high. We’re working with closing prices.

Figure 6: Series plot

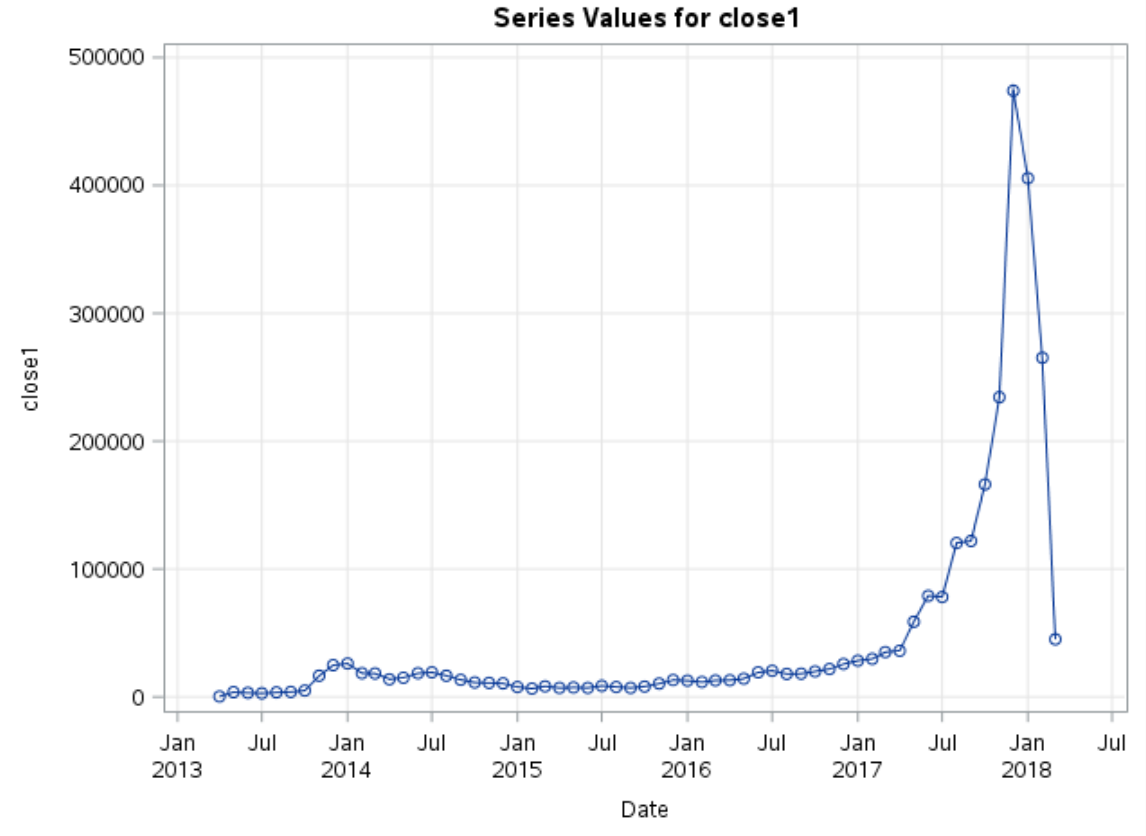
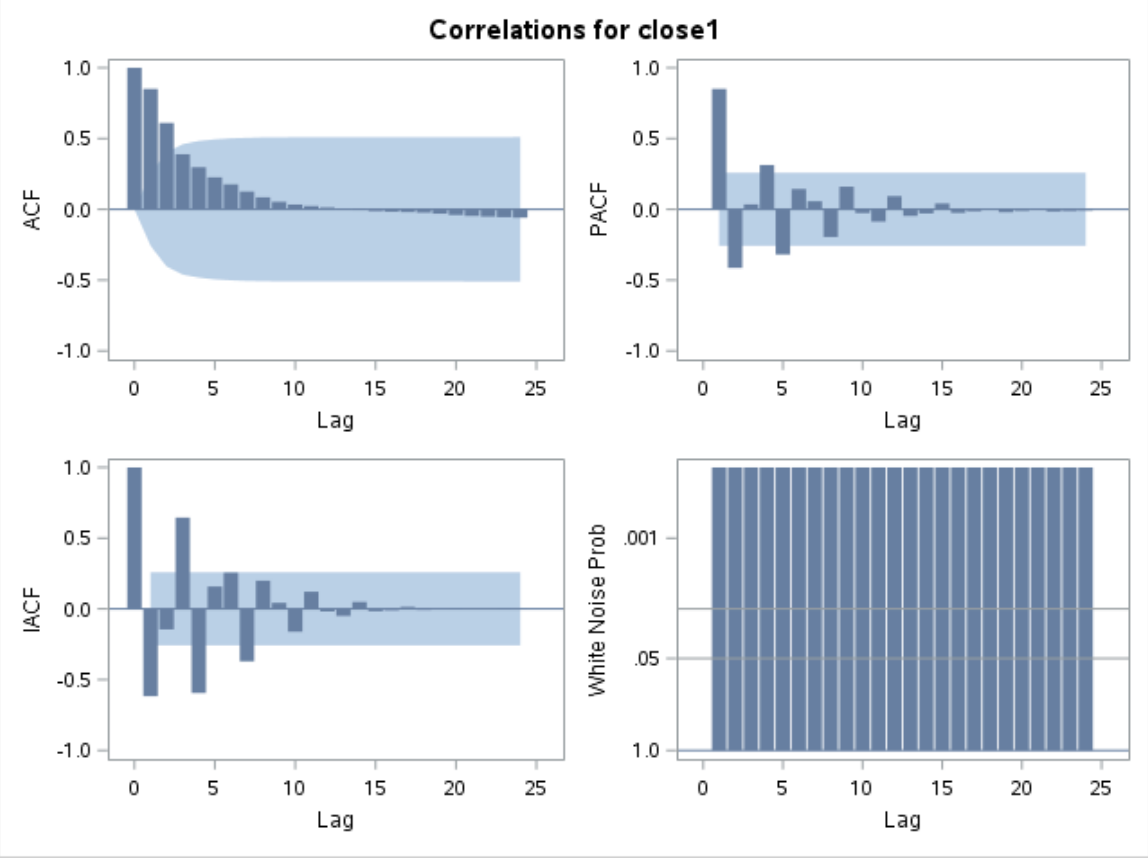
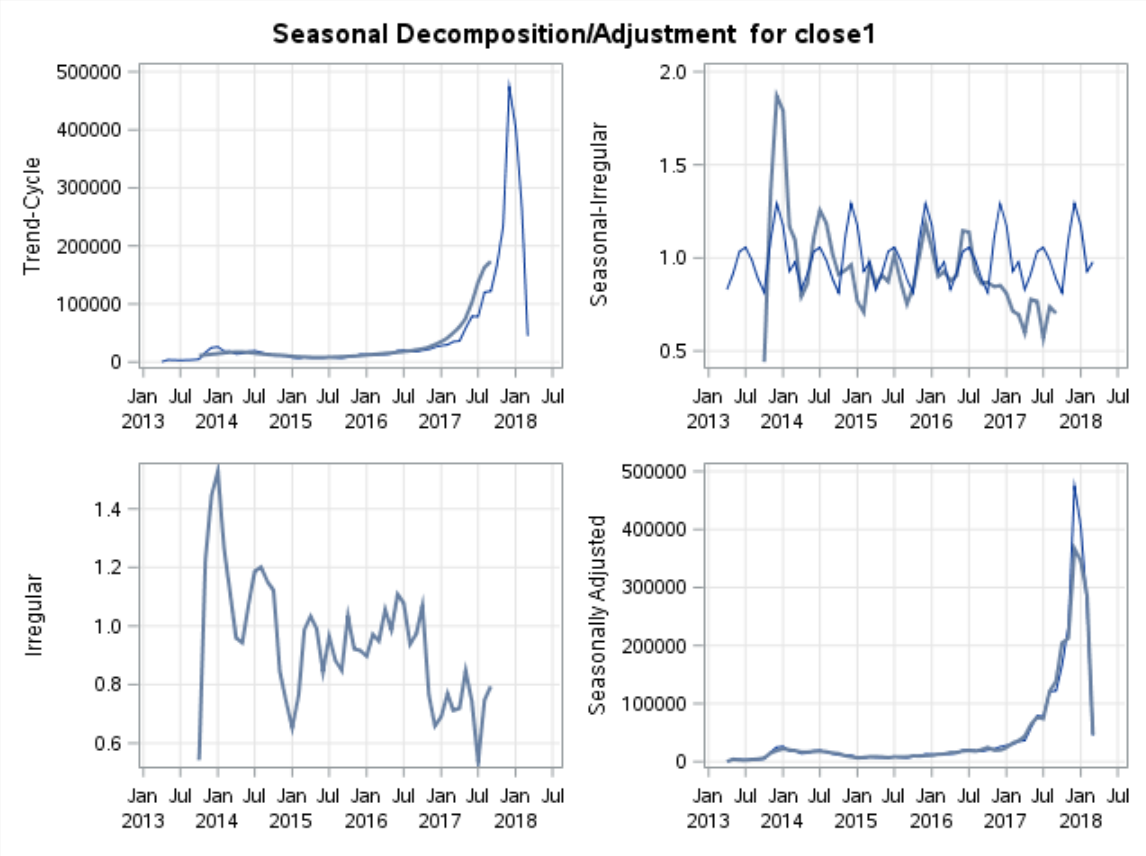


Figure 7: Correlations plot



The autocorrelation plot in the top left is a bar chart of the coefficients of correlation between a time series and lags of itself. The partial *auto*correlation (top right) is the amount of correlation between a variable and a lag of itself that is not explained by correlations at all *lower-order*-lags[[7]](#footnote-6). The inverse of the ACF is the IACF. for the white noise probability plot (bottom right) the longer bars favor rejection of the null hypothesis that the prediction errors represent white noise. In this example, they are all significant beyond the 0.001 probability level, so that you reject the null hypothesis.[[8]](#footnote-7)

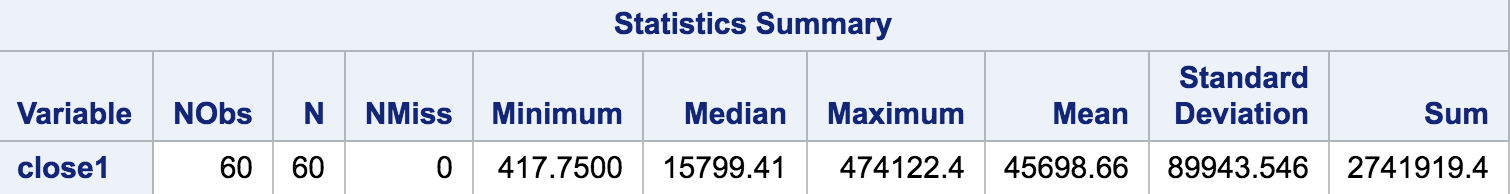
Figure 8: Seasonal decomposition plot



Seasonally adjusted plot (bottom right) looks very similar to my series plot in figure 6.

The seasonal-irregular plot (top right) shows much noise. This makes it hard to spot seasonal trends.

Table 8: SAS summary statistics



Proc timeseries used for market\_cap1 variable. Bitcoin1 is the output data set. I have asked for trend and seasonal analysis as well as a series plot.

PROC TIMESERIES DATA=bit3

OUT=bitcoin1

outtrend=trend1 print = trends

OUTSEASON=bit81 print = seasons;

ID Date INTERVAL=month

ACCUMULATE=total;

VAR market\_cap1;

RUN;

Table 9: SAS trend statistics

Have statistics for sixty months (five years of data). Only 5 shown.

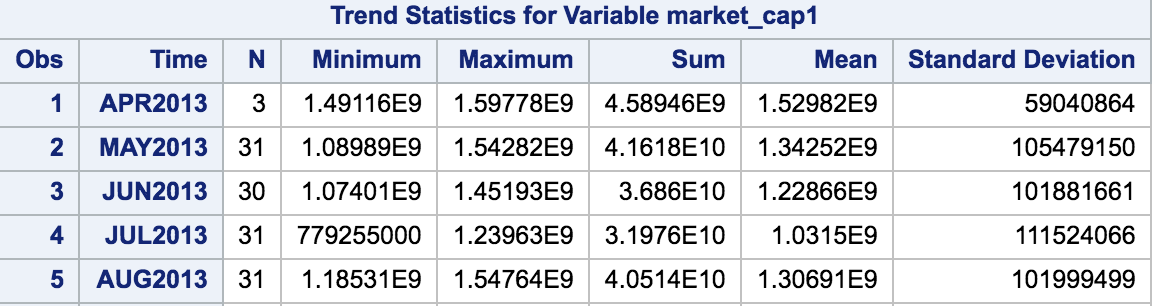


Table 10: SAS global statistics

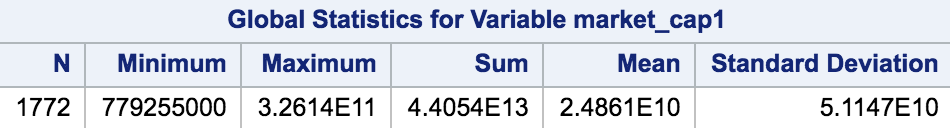
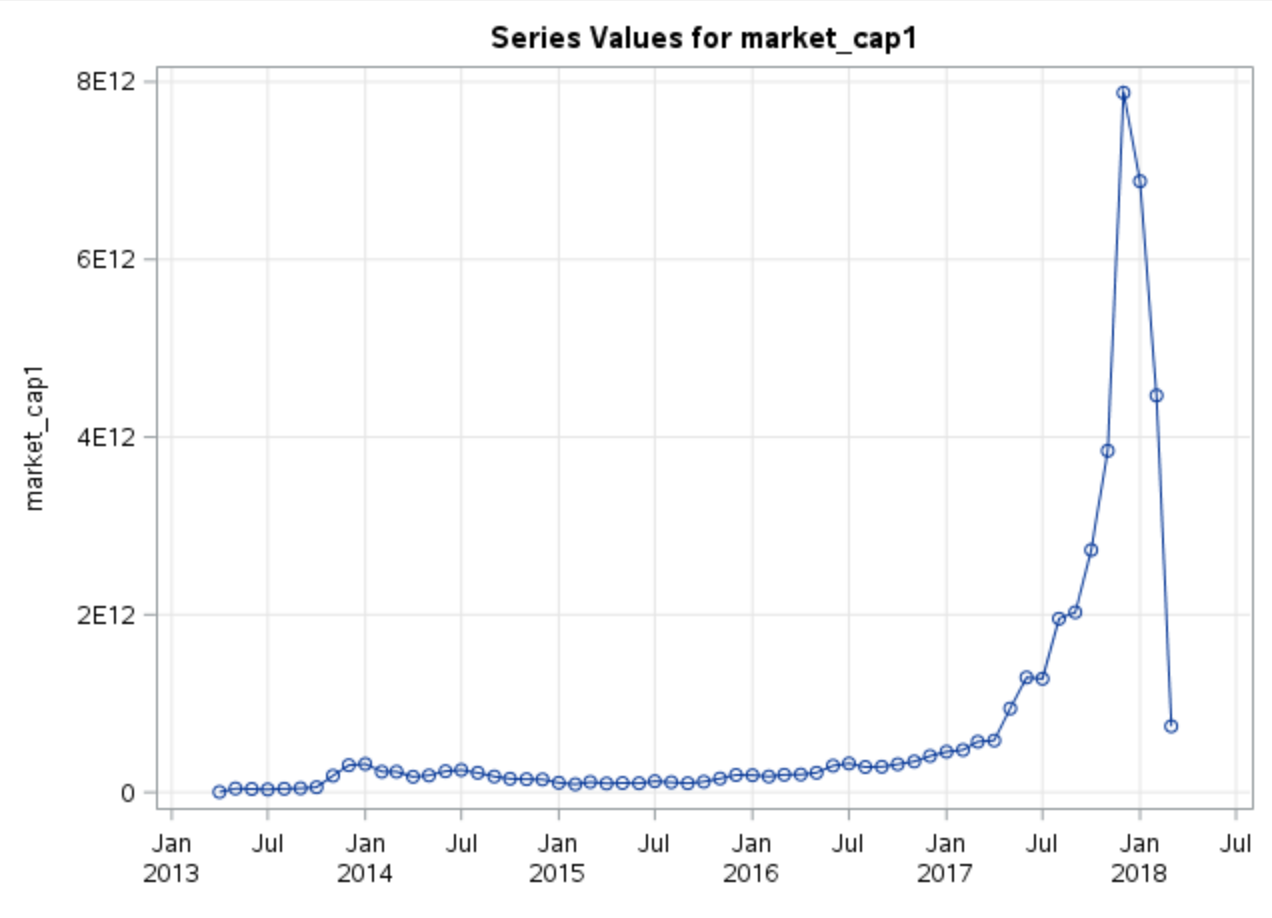


Figure 9: Series plot

Notice the similarities with the series plot for close1 (the closing price).



The data set “trend” is trend data stored from close1.

title1 "Max,Mean and Min per month overlayed";

proc sgplot data=trend;

series x=date y=max / lineattrs=(pattern=solid);

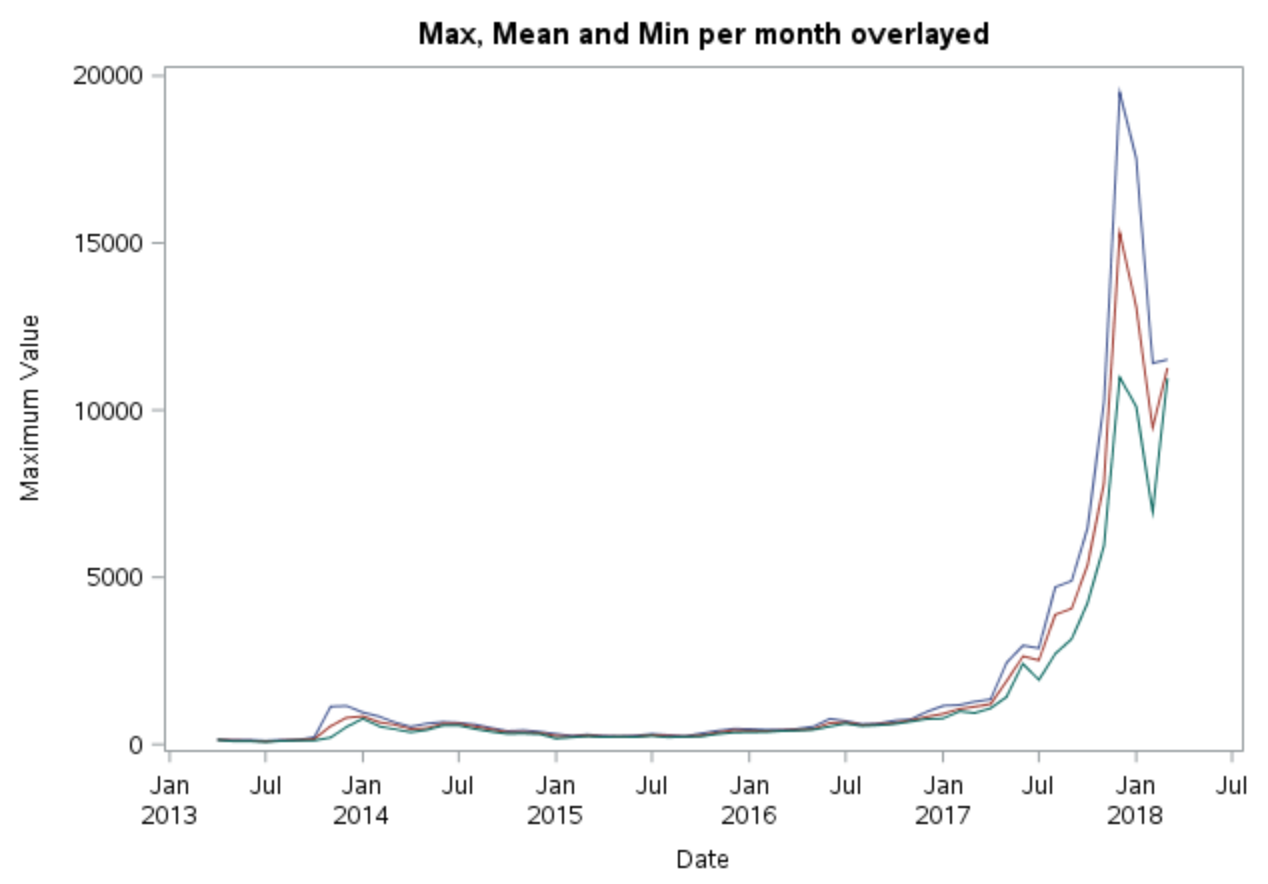
series x=date y=mean / lineattrs=(pattern=solid);

series x=date y=min / lineattrs=(pattern=solid);

run;

Figure 10: Trend statistics graph

The graph is a visual look at monthly volatility.



Similarly done with market\_cap1.

title3 "trend of marketcap";

proc sgplot data=trend1;

series x=date y=max / lineattrs=(pattern=solid);

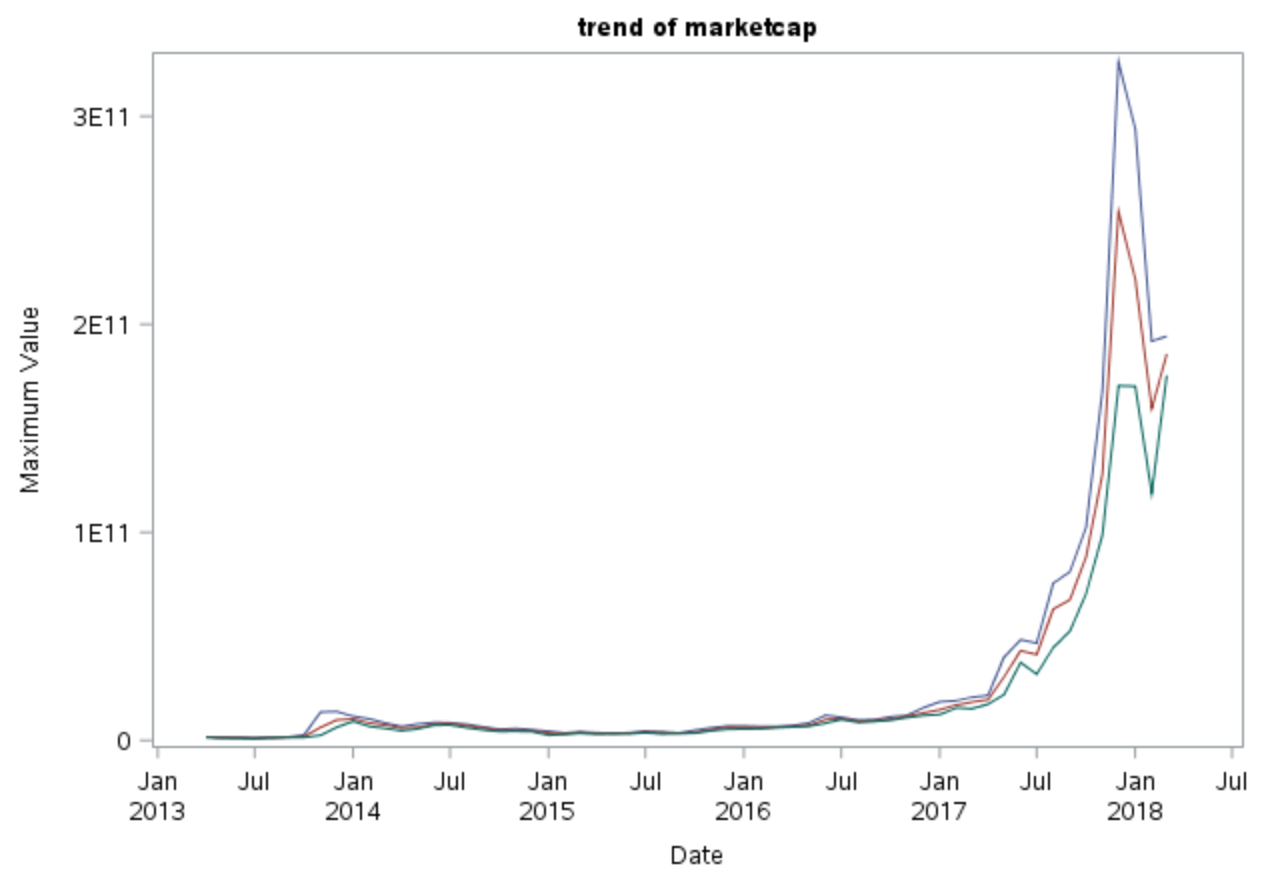
series x=date y=mean / lineattrs=(pattern=solid);

series x=date y=min / lineattrs=(pattern=solid);

/\* yaxis display=(nolabel); \*/

Run;

Figure 11: Trend statistics graph



Looking at a vertical bar plot for close1. Bars represent the monthly mean.

title2 "Mean close per month";

proc sgplot data=trend;

vbar date / freq=mean;

/\* XAXIS FITPOLICY=THIN; \*/

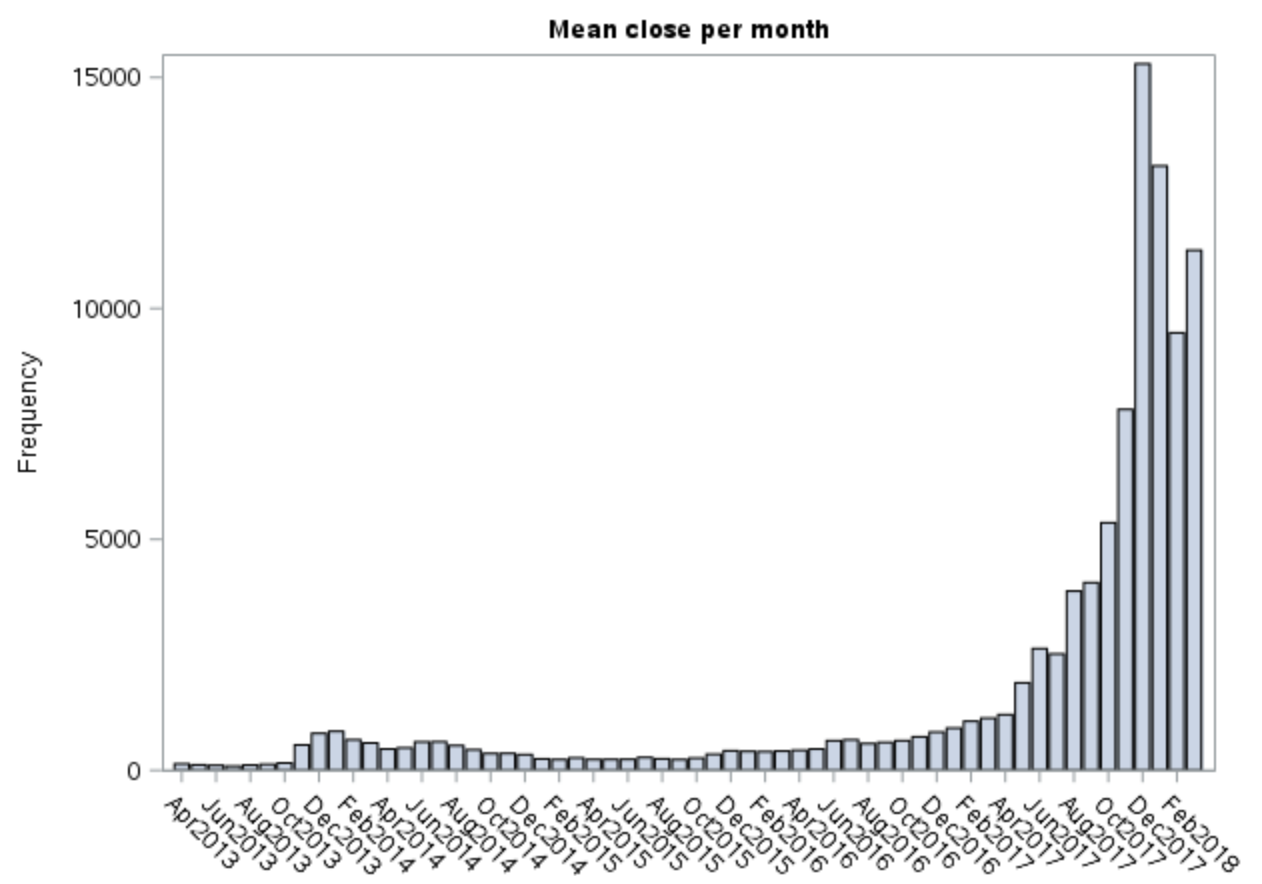
XAXIS FITPOLICY= ROTATETHIN;

/\* XAXIS FITPOLICY=STAGGER; \*/

/\* format date MONYY7.; \*/

run;

Figure 12: Bar chart



Trend statistics examined by year as well. Trend data set was set into trend2 and made date look at year instead of month.

data trend2;

set trend;

year = year(date);

run;

proc print data = trend2;

title1 "Trend Statistics by Year";

proc sgplot data=trend2;

series x=\_season\_ y=mean / group=year lineattrs=(pattern=solid);

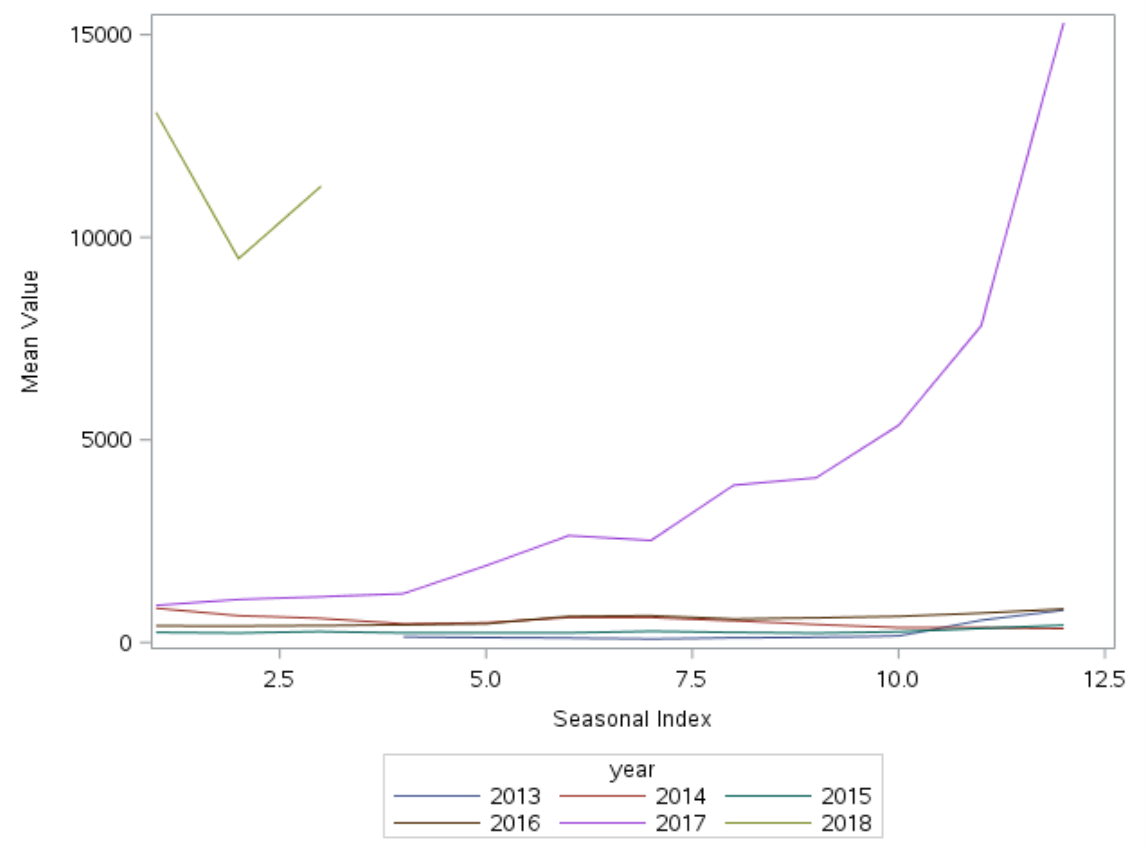
/\* xaxis values=(1 to 12 by 1); \*/

run;

PROC PRINT DATA = TREND2;

RUN;

Figure 13: Seasonality graph



Seasonal index for the max, mean and min. The seasonal index represents the twelve months for each year. Bit 8 is the outseason data set for close1. The graph shows the huge spike in bitcoin prices for the year 2017. Notice we only have data for the first three months of 2018.

proc sgplot data=bit8;

series x=\_season\_ y=max / lineattrs=(pattern=solid);

series x=\_season\_ y=mean / lineattrs=(pattern=solid);

series x=\_season\_ y=min / lineattrs=(pattern=solid);

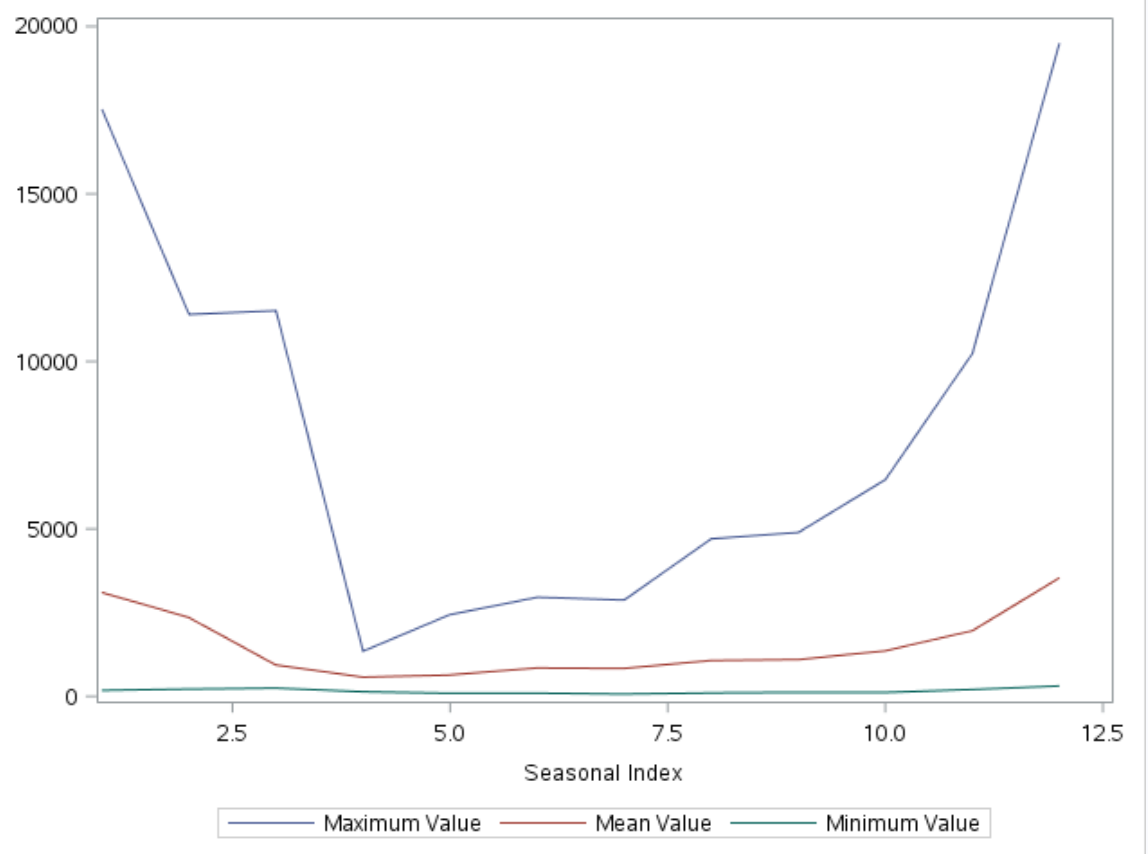
yaxis display=(nolabel);

/\* xaxis values=(1 to 4 by 1); \*/

run;

Clearly the maximum price values had the greatest fluctuation in a given month.

Figure 14: Seasonality graph



Cross series plot to look at market\_cap and close1 on the same graph with each scale.

title "Illustration of ODS Graphics";

proc timeseries data=bit3 out=\_null\_

plots=(series corr decomp)

crossplots=all;

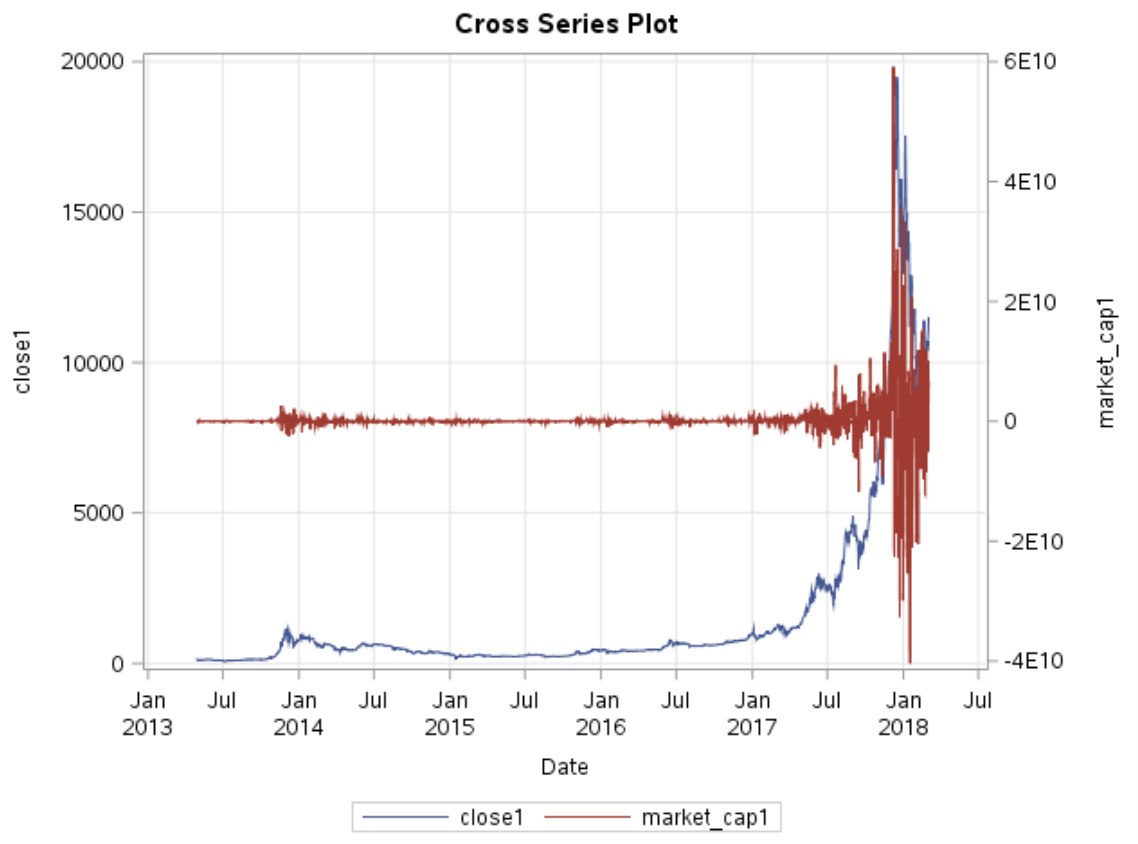
id date interval=day;

var close1;

crossvar market\_cap1 / dif=(1);

run;

Figure 15: Cross series plot



# Market\_cap and close1 both are steady until around July of 2017. They both spiked at the same time. This shows that as price increase the quantity of bitcoin mined likely follows.

# **Conclusion**

By working with stock pricing data from Standard and Poor’s 500 and Bitcoin data the application of PROC TIMESERIES was evidenced. PROC TIMESERIES is fairly robust and simple yet powerful. PROC TIMESERIES allows for much creativity and freedom when it comes to adding detail and illustration in graphical displays.

PROC TIMESERIES lends itself well in creating informative graphs to visualize key characteristics of time series data, such as trend and seasonal statistics and autocorrelation. As the last example illustrated, the procedure can be utilized to examine variable relationships (cross series plot), taking it a step farther than an analysis of one variable versus time.

Understanding the trends and cycles in historical data is crucial for some analysis, but another goal when dealing with time series data may be to predict the future behavior of an industry, perhaps retail demand or stock prices. While PROC TIMESERIES is very powerful on its own it is often utilized in conjunction with PROC FORECAST, PROC SPECTRA, and PROC ARIMA. These procedures can produce mathematical models that can then be used for prediction purposes.[[9]](#footnote-8)

# **References**

Gharibvand, Somaye; Gharibvand, Lida. 2010. “Financial Analysis Using SAS ® PROCS.” *Proceedings of the SAS Global 2010 Conference.* Cary, NC: SAS Institute, Inc. Available at:<https://support.sas.com/resources/papers/proceedings10/261-2010.pdf>

Department of Statistics Online Programs. “STAT 510 | Applied Time Series Analysis.” *Pennsylvania State Univerity*. 17 March 2018. Available at:<https://onlinecourses.science.psu.edu/stat510/node/47>

SAS Institute, Inc. (2008), SAS/ETS(R) 9.2 User's Guide TIMESERIES Procedure Functional Summary, SAS OnlineDoc® 9.2, Cary, NC: SAS Institute, Inc. Available at:<http://support.sas.com/documentation/cdl/en/etsug/60372/HTML/default/timeseries_toc.htm>

SAS Institute, Inc. (2008), SAS/ETS(R) 9.3 User's Guide Examples: TIMESERIES Procedure, SAS OnlineDoc® 9.3, Cary, NC: SAS Institute, Inc. Available at:<http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm#etsug_timeseries_sect042.htm>

Xing, Juan. “Application of Using SAS for Time Series Forecasting.” *sas.com.* 2015. Available at:<https://www.sas.com/content/dam/SAS/en_ca/User%20Group%20Presentations/Winnipeg-User-Group/Xing-TimeSeriesForecasting-Fall2015.pdf>

SAS Institute, Inc. (2008), SAS/ETS(R) 9.3 User's Guide Examples: Model Viewer, SAS OnlineDoc® 9.3, Cary, NC: SAS Institute, Inc. Available at:

[http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm#etsug\_tfstart\_sect025.htm](http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm%23etsug_tfstart_sect025.htm)

SAS Institute, Inc. (2008), SAS/ETS(R) 13.2 User's Guide Examples: The TIMESERIES Procedure, SAS OnlineDoc® 13.2, Cary, NC: SAS Institute, Inc. Available at:

<https://support.sas.com/documentation/onlinedoc/ets/132/timeseries.pdf>

Nau, Robert. “Arima models for time series forecasting.” Duke.edu. 2017. Available at:

<http://people.duke.edu/~rnau/411arim3.htm>

SAS Institute, Inc. (2008), SAS/ETS(R) 9.2 User's Guide Examples: TIMESERIES Procedure, SAS OnlineDoc® 9.2, Cary, NC: SAS Institute, Inc. Available at:

<http://support.sas.com/documentation/cdl/en/etsug/60372/HTML/default/viewer.htm#etsug_timeseries_sect002.htm>

SAS Institute, Inc. (2008), SAS/ETS(R) 13.1 User's Guide Examples: The TIMESERIES Procedure, SAS OnlineDoc® 13.1, Cary, NC: SAS Institute, Inc. Available at:

<http://support.sas.com/documentation/cdl/en/etsug/66840/HTML/default/viewer.htm#etsug_timeseries_gettingstarted.htm>

SAS Institute, Inc. (2008), SAS/ETS(R) 13.1 User's Guide Examples: The TIMESERIES Procedure, SAS OnlineDoc® 13.1, Cary, NC: SAS Institute, Inc. Available at:

<http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm#etsug_timeseries_sect044.htm>

1. https://support.sas.com/resources/papers/proceedings10/261-2010.pdf [↑](#footnote-ref-0)
2. https://onlinecourses.science.psu.edu/stat510/node/47 [↑](#footnote-ref-1)
3. http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm#etsug\_timeseries\_sect005.htm [↑](#footnote-ref-2)
4. https://support.sas.com/resources/papers/proceedings10/261-2010.pdf [↑](#footnote-ref-3)
5. http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm#etsug\_timeseries\_sect044.htm [↑](#footnote-ref-4)
6. http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm#etsug\_timeseries\_sect046.htm [↑](#footnote-ref-5)
7. http://people.duke.edu/~rnau/411arim3.htm [↑](#footnote-ref-6)
8. http://support.sas.com/documentation/cdl/en/etsug/63939/HTML/default/viewer.htm#etsug\_tfstart\_sect025.htm [↑](#footnote-ref-7)
9. https://www.sas.com/content/dam/SAS/en\_ca/User%20Group%20Presentations/Winnipeg-User-Group/Xing-TimeSeriesForecasting-Fall2015.pdf [↑](#footnote-ref-8)