Paper AA11-2015

Shifting Expectations: A practical algorithm for detecting level shifts using robust rolling statistics

Matt Bates, J.P. Morgan Chase, Columbus, OH

ABSTRACT

In the world of predictive analytics, identifying level shifts in the data is like flying an airplane. The current observed altitude (level) generally contains more relevant information about the near future- and if not given proper attention the flight may end up crashing through a mountain. It is a sound practice of a forecaster to use level shift information to determine the ideal size for the training data, so long as seasonality and/or trending cannot be safely assumed. The proposed method to be discussed is intended for practitioner predictive modelers with a basic understanding of level shifts. All demonstrations are executed using SAS 9.2 and only requires SAS/STAT® software (SAS/ETS® software is not needed).

INTRODUCTION

In the world of forecasting temporal data, one of the most crucial (and often very subjective) decisions any analyst must make is choosing the "correct" time period to allow in a training data set for a forecast model. Many analysts feel that more data is better, and in cases where seasonality is strong (similar events that happened multiple times in a regular pattern) or there is a clear trend of any type that can be assumed reasonably to continue on, then this philosophy is good. However when something occurs in the data that is fundamentally different then prior data, using all the data may result in a worse prediction. Consider the Figure 1.1 below:

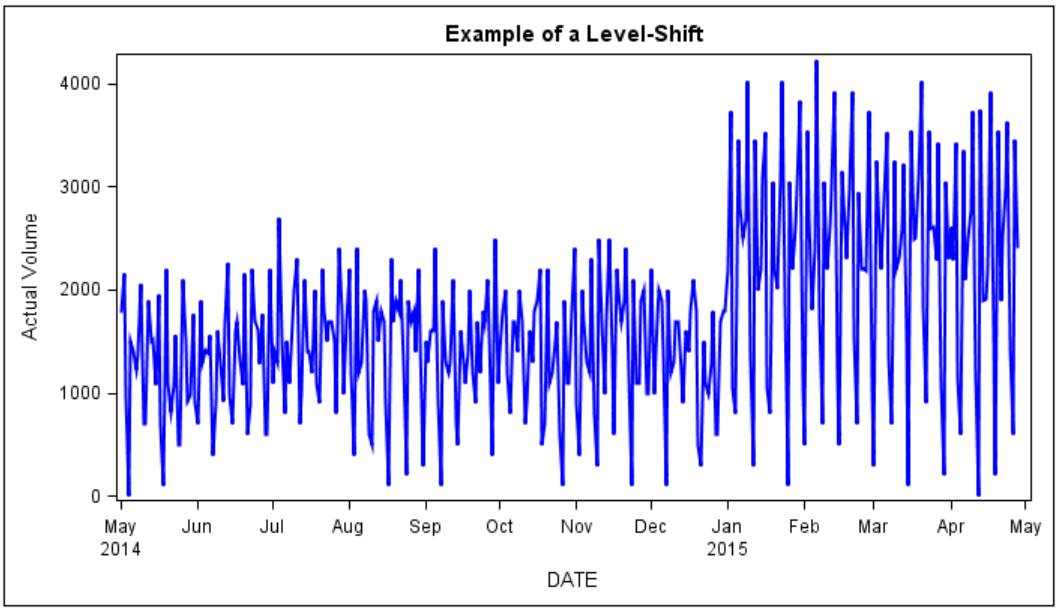

Figure 1.

If all the data were used for forecasting, predictions beyond April of 2015 would mostly likely be too low because data in 2014 would influence estimates to be low relative to the closer time period in 2015. Therefore it makes sense that only data from January 2015 going forward should be used in a training data. It is a common practice among forecasters to choose the time period based on such a chart and is generally taught in most regression courses as an encouraged practice to improve forecast models. What is not as often taught to practitioner predictive modelers is how these time periods can be chosen

algorithmically, both to eliminate the subjectivity of a particular forecaster as well as provide a means for automating forecasts-particularly when there are too many entities for a single forecaster to make specific decisions for on each forecast model.

The inspiration in the development of the proposed technique for level shift detection comes from the challenge to forecast the daily withdrawal amount for thousands of ATMs on a regular basis for multiple reasons. In this context, all data presented is artificially generated from SAS data steps for pedantic purposes, however the nature of this artificial data is very realistic to forecasting daily withdrawals for typical ATMs. Often new businesses open up or close and the result there is often significant level shifts in the data. For this reason any "by" processing demonstrating different models for different entities will be done using the entity of "ATM".

Overview of the Rolling Statistic algorithm:

A common classification of level shift algorithms are known as "Sliding Window" algorithms because the identification of level shifts depends upon a "window" that is moved across the data. The proposed algorithm being discussed is a sliding window and relies upon observing the changes in robust statistics calculated from each window. More specifically the algorithm uses a window of 15 (but can easily be modified) and calculates the 1st 2nd and 3rd quartile as well as a trimmed mean. Comparison between each window and the neighboring windows is made to see if there is a consistent significant shift in the majority of these statistics.

Calculating rolling statistics can be cumbersome and thought to be near impossible or impractical to code by most programmers who are unaware the power of the multi-level formatting capabilities of SAS. The multi-level format option is what makes this algorithm both possible and practical as well.

What is a multi-level format?

A multi-level format is a format that when used in conjunction with certain procedures such as proc means and allows a single record to be represented in more than one values of the output. It is usually built by starting with a data set that has a distinct list of values and adding a start and end field based on that value. This data set is then read into the format library for use in downstream procedures. As an example if you had a data set containing 31 records for each day in March a format that could be created to produce a rolling average a data set would be created to look like Figure 1.2 (only first 5 records shown):

```
DATA FMT;*DEFINE A FORMAT FOR THE ROLLING WINDOW OF STATISTICS;
       RETAIN FMTNAME "ROLL"  TYPE 'N'  HLO 'M';
       SET DISTDATE(RENAME=(DATE=START));
       FORMAT START END DATE.;
       END=START+5-1; *+5-1 EMPHASIZING A ROLLING WINDOW OF SIZE 5;
       LABEL=PUT(START+2,DATE.); *+2 ADDED SO THE RESULTS WILL REFLECT THE MIDDLE OF
THE WINDOW OF SIZE 5;
       OUTPUT;
RUN;

PROC FORMAT CNTLIN=FMT;RUN;
```

VIEWTABLE: Work.Fmt

	FMTNAME	TYPE	HLO	START	END	LABEL
1	ROLL	N	M	01MAR15	05MAR15	03MAR15
2	ROLL	N	M	02MAR15	06MAR15	04MAR15
3	ROLL	N	M	03MAR15	07MAR15	05MAR15
4	ROLL	N	M	04MAR15	08MAR15	06MAR15
5	ROLL	N	M	05MAR15	09MAR15	07MAR15
6	ROLL	N	M	06MAR15	10MAR15	08MAR15

Figure 2.

In this example if the name of the format is called using the multi-level format within a means procedure then the record of "05Mar15" would be used in the average for records 1, 2, 3, 4 and 5 because the date is between the start and end date for those 5 records. To calculate the average the proc means procedure needs to call the multi-level format using the class variable with the options of preloadfmt mlf exclusive like the following:

VIEWTABLE: Work.Volume

	date	usage
1	01MAR15	11
2	02MAR15	18
3	03MAR15	3
4	04MAR15	3
5	05MAR15	7
6	06MAR15	18
7	07MAR15	4
8	08MAR15	31
9	09MAR15	1

Figure 3.

```
*COMPUTE ROLLING AVERAGES;
PROC MEANS DATA=VOLUME NOPRINT;
        CLASS DATE /PRELOADFMT MLF EXCLUSIVE;
        FORMAT DATE ROLL.;
        VAR USAGE;
        OUTPUT OUT=ROLLING_MEAN MEAN=ROLLING_MEAN;
RUN;
```

VIEWTABLE: Work.Rolling_mean

	date	_TYPE_	_FREQ_	ROLLING_MEAN
1		0	31	9.064516129
2	01APR15	1	2	6.5
3	02APR15	1	1	10
4	03MAR15	1	5	8.4
5	04MAR15	1	5	9.8
6	05MAR15	1	5	7
7	06MAR15	1	5	12.6
8	07MAR15	1	5	12.2
9	08MAR15	1	5	12.4
10	09MAR15	1	5	10

Figure 4.

The resulting output data set converts the date column from numeric to character in this process. The first record with date ="" is the average of all 31 records. Records 2 and 3 are the result of the format declaring the label to be the date + 2 for the purpose of returning the center of each average record. The _FREQ_ emphasizes how many records where used to compute that average. As a result it may be a good practice to add a where statement to the output statement like:

```
        OUTPUT OUT=ROLLING_MEAN(WHERE=( _FREQ_=5 )) MEAN=ROLLING_MEAN;
```

Since the resulting date column was converted to a character value it may be suggested to also reconvert back to a numeric column in a follow up data set like:

```
DATA ROLLING_MEAN;*EXTRACT DATE FROM TEXT;
        FORMAT DATE DATE.;
        SET ROLLING_MEAN(RENAME=(DATE=DATE_CHAR));

        *EXTRACTING THE NUMERIC DATE FROM A TEXT FIELD;
        DATE=INPUT(DATE_CHAR,DATE.);
        DROP DATE_CHAR;
RUN;
```

VIEWTABLE: Work.Rolling_mean

	DATE	_TYPE_	_FREQ_	ROLLING_MEAN
1	03MAR15	1	5	8.4
2	04MAR15	1	5	9.8
3	05MAR15	1	5	7
4	06MAR15	1	5	12.6
5	07MAR15	1	5	12.2
6	08MAR15	1	5	12.4

Figure 5.

Robust Rolling Statistics and Window Size:

Outliers can often severely affect models, even moderate ones, and when using only a few records of data to calculate statistics, even minor outliers may cause erroneous results. For this reason, robust type statistics are desired to drive decisions on perceived significance of a jump or fall in the statistics. Therefore 4 robust statistics were chosen for this algorithm: 1st , 2nd, 3rd quartile and a trimmed mean. The trimmed mean in this instance is only trimming the minimum and maximum values and is calculated as follows: (sum-min-max)/(n-2).

The use of robust statistics in the proposed algorithm is particularly critical given the recommend window size of 15. Although there is nothing magical about the window size a user needs to realize that the selection of the window size in analogous to the Type 1 and Type 2 errors in hypothesis testing. Make the window too big and you will fail to detect big levels shifts that should have been detected. Make the window to small and every bump may appear like a level shift when there is none. The window size of 15 was chosen through much trial an error by observing graphical results along with other factors on many different sets of data. It is recommended, however, for each user to experiment with this size for their given data to see if improvements can be obtained.

Discuss methodology of declaring a level shift:

The philosophy as adopted for the rolling statistic algorithm is to not declare a level shift unless there is strong enough evidence in the results to suggest a new level. Therefore a given set of data may have a moderate level shift and this algorithm might suggest the use of all data even though visually a forecaster may have made a different decision. It is preferable not to throw data away unless there is confidence in the algorithm that throwing away the old data is truly warranted. With this in mind the following steps are executed to identify possible level shifts.

Step 1) Rolling Statistics data set is sorted by ATM, Weekday and descending date.

Step 2) In 3 separate respective columns determine the next 3 weeks statistics using a retain statement (ie. if the date is a November 3rd – a Monday- then November 10th,17th, 24th (all future Mondays from November 3rd) values for the rolling statistics will be saved off in the same November 3rd record

Step 3) For each of the 4 Rolling statistics compute the ratio of the current statistic with 2 and 3 weeks out (ie. the ratio of the 1st Rolling Quartile on November 3rd will be divided by the 1st Rolling Quartile of November 17th. A second ratio is also calculates as the 1st Rolling Quartile on November 3rd divided by 1st Rolling Quartile of November 24th.)

Step 4) If 3 out the 4 statistics are both consistently high (or both consistently low) for both comparisons to week 2 and week 3 in the future then output as a possible level shift.

Step 5) Excluding any possible level shifts that are within 34 days of the end of the training data select the latest level shift as the declared time period to use going forward.

Refer to the yellow highlight portion of the macro definition below to see how the 5 steps are carried out.

```
%MACRO LEVEL_SHIFTS(INDATA,OUTDATA,ID,DATE,VOL,LATEST_DATE='30APR15'D-
50,WINDOW=15,CHANGE_FACTOR=.4);
PROC SQL;*CREATE TABLE OF DISTINCT DATES- FOR MULTI-FORMAT (ROLLING WINDOW) TECHNIQUE;
      CREATE TABLE DISTDATE AS SELECT DISTINCT &DATE. AS START FORMAT=DATE. FROM &INDATA.;
QUIT;

DATA FMT;*DEFINE A FORMAT FOR THE ROLLING WINDOW OF STATISTICS;
      RETAIN FMTNAME "ROLL"  TYPE 'N'  HLO 'M';
      SET DISTDATE;
      FORMAT START END DATE.;
      END=START+&WINDOW.-1;
      LABEL=PUT(END,DATE.);
      OUTPUT;
RUN;

PROC FORMAT CNTLIN=FMT;RUN;

PROC SORT DATA=&INDATA.;BY &ID.;RUN;

PROC MEANS DATA=&INDATA. NOPRINT;*COMPUTE ROLLING STATISTICS;
      BY &ID.;
      CLASS &DATE. /PRELOADFMT MLF EXCLUSIVE;
      FORMAT &DATE. ROLL.;
      VAR &VOL.;
      OUTPUT OUT=ROLL_STAT(DROP=_TYPE_ WHERE=(_FREQ_=&WINDOW.)) Q1=Q1 Q3=Q3 MEDIAN=MEDIAN
MAX=MAX MIN=MIN SUM=SUM;
RUN;

DATA ROLL_STAT;*EXTRACT DATE FROM TEXT,CALCULATE TRIMMED ROLLING MEAN;
      SET ROLL_STAT;
      *CALCULATING A TRIMMED MEAN;
      TMEAN=(SUM-MAX-MIN)/(_FREQ_-2);
      FORMAT LEVEL_DATE DATE.;
      *EXTRACTING THE NUMERIC DATE FROM A TEXT FIELD;
      LEVEL_DATE=INPUT(&DATE.,DATE.);
      WKDY=WEEKDAY(LEVEL_DATE);
      DROP SUM MAX MIN _FREQ_ &DATE.;
RUN;

PROC SORT DATA=ROLL_STAT;BY &ID. WKDY DESCENDING LEVEL_DATE;RUN;

*OUTPUTS EVERY DETECTED POSSIBLE LEVEL SHIFT BY COMPARING EVERY OTHER PERIOD;
DATA LEVEL_SHIFTS7(WHERE=(LEVEL_DATE LE &LATEST_DATE.+ROUND(&WINDOW./2,1))
KEEP=&ID. WKDY LEVEL_DATE);
      SET ROLL_STAT;
      BY &ID. WKDY;
      RETAIN       PREV_Q1 PREV_Q3 PREV_MED PREV_TMEAN
                   PREV_Q12 PREV_Q32 PREV_MED2 PREV_TMEAN2
                   PREV_Q13 PREV_Q33 PREV_MED3 PREV_TMEAN3;

      FORMAT DIFF_Q12 DIFF_Q32 DIFF_MED2 DIFF_TMEAN2
                DIFF_Q13 DIFF_Q33 DIFF_MED3 DIFF_TMEAN3 PERCENT6.0;

      *DEFINING ARRAYS FOR EASE OF CODING;
      ARRAY P{12} PREV_Q1--PREV_TMEAN3;
      ARRAY S{4} Q1 Q3 MEDIAN TMEAN;
      ARRAY D{8} DIFF_Q12--DIFF_TMEAN3;

      IF FIRST.&ID. OR FIRST.WKDY THEN DO I=1 TO 12;P(I)=.;END;

      *RATIO BETWEEN 1ST 2ND AND 3RD WEEK OUT;
      *DETERMINE NUMBER OF STATISTICS WERE SIGNIFICANTLY HIGH OR LOW;
            HIGH2=0;HIGH3=0;LOW2=0;LOW3=0;
```

```
        DO I=1 TO 4;
             *ENSURE THAT ZEROS AND MISSING ARE NOT DRIVING RESULTS;
             IF P(I+4) NOT IN(.,0) and P(I+8) NOT IN(.,0) AND S(I) NOT IN(.,0) THEN DO;
             *COMPARING TO TWO WEEKS OUT;
             D(I)=S(I)/P(I+4);
             IF D(I)>(1+&CHANGE_FACTOR.) THEN HIGH2=HIGH2+1;
             IF 1/D(I)>(1+&CHANGE_FACTOR.) THEN LOW2=LOW2+1;
             *COMPARING TO THREE WEEKS OUT;
             D(I+4)=S(I)/P(I+8);
             IF D(I+4)>(1+&CHANGE_FACTOR.) THEN HIGH3=HIGH3+1;
             IF 1/D(I+4)>(1+&CHANGE_FACTOR.) THEN LOW3=LOW3+1;
             END;
             *USING RETAIN FEATURE TO BRING PREVIOUS VALUES INTO NEXT RECORDS;
             P(I+8)=P(I+4);
             P(I+4)=P(I);
             P(I)=S(I);
             END;
        DROP I;

        *DECLARE LEVEL SHIFT IF 3 OUT OF 4 STATISTICS ARE SIGNIFICANT 2 WEEKS IN A ROW;
        IF (HIGH2>2 AND HIGH3>2) OR (LOW2>2 AND LOW3>2) THEN IND=1;

        IF IND=1 THEN OUTPUT;
RUN;

PROC SORT DATA=LEVEL_SHIFTS7;BY &ID. LEVEL_DATE;RUN;

DATA &OUTDATA.;*OUTPUT ONLY THE LATEST DETECTED LEVEL SHIFT;
        SET LEVEL_SHIFTS7;
        BY &ID.;
        *CHANGING TO BE ROUGHLY THE MIDPOINT OF THE ROLLING STATISTICS;
        FORMAT LEVEL_DATE DATE.;
        LEVEL_DATE=LEVEL_DATE+ROUND(&WINDOW./2,1);
        IF LAST.&ID. THEN OUTPUT;
RUN;
%MEND;
```

Testing Results:

Testing this level shift macro with different window lengths and change factor values was applied to known ATMs that had significant level shifts. To expand this testing a data step was written to create artificial code with purposeful level shifts from 01May2014 to 30Apr2015 and included a “ramp up period” where the first 34 days were 30% lower then the typical days. The shift for all artificial ATMs was set to occur on 01Jan2015 in steps up of 50%,75% and 100% increase as well as 30%, 50%, 70% decrease periods from typical values. Weekday factors were included with random uniform errors as well as random normal errors were introduced. With the seed is set to “0” each execution provided different errors, and thus different results. This was done on a few different times to get a “feel” for how well the rolling algorithm performed comparing to the visual inspection of the charts. Two conclusions out of this testing that be came apparent is that the algorithm does not capture the shift that really did occur every time due to random patterns- but did appear to capture an appropriate cutoff more than 70% of the time for the degree of level shifts that were specified in the underlying data. The second conclusion is that the algorithm often picks up appropriate level shifts in the data when visually the shift was not apparent- adding to the value of using this algorithm in forecasting.

Data set Generator of ATMS with Level Shifts:

```
%LET SEED=123;
*DEFINING LIST OF 12 ATMS;
DATA ATMS;FORMAT ATM $27.;INPUT ATM $;DATALINES;
ATM1_50UP_UNIFORM_ERROR
ATM2_75UP_UNIFORM_ERROR
ATM3_100UP_UNIFORM_ERROR
ATM4_30DOWN_UNIFORM_ERROR
ATM5_50DOWN_UNIFORM_ERROR
ATM6_70DOWN_UNIFORM_ERROR
ATM7_50UP_NORMAL_ERROR
ATM8_75UP_NORMAL_ERROR
ATM9_100UP_NORMAL_ERROR
ATM10_30DOWN_NORMAL_ERROR
ATM11_50DOWN_NORMAL_ERROR
ATM12_70DOWN_NORMAL_ERROR
;
RUN;

DATA ATMDATA;*CREATING THE SIMULATED ATM DATA WITH VARYING DEGRESS OF LEVEL SHIFTS;
SET ATMS;
FORMAT DATE DATE.;
*CREATING 1 YEAR OF RECORDS;
DO DATE='30APR15'D-364 TO '30APR15'D;
W=WEEKDAY(DATE);

*CREATING ARTIFICIAL WEEKDAY FACTORS;
IF W=1 THEN FACT=.3;
ELSE IF W=2 THEN FACT=1.5;
ELSE IF W=3 THEN FACT=1;
ELSE IF W=4 THEN FACT=.9;
ELSE IF W=5 THEN FACT=1.1;
ELSE IF W=6 THEN FACT=1.7;
ELSE IF W=7 THEN FACT=.5;

USAGE=1000*FACT;

DROP W FACT;

*CREATING RANDOM ERRORS FOR ATMS- UNIFORM FOR FIRST 7 NORMAL FOR LAST 6;
IF FIND(ATM,"UNIF")>0 THEN ERR=1000*RANUNI(&SEED.);
ELSE ERR=1000*ABS(RANNORM(&SEED.));

*CREATE ARTIFICIAL LEVEL SHIFTS;
IF DATE <'30APR15'D-330 THEN USAGE=USAGE*.7;
ELSE IF DATE GE '01JAN15'D THEN DO;
IF _N_ IN(1,7) THEN USAGE=USAGE*1.5;
IF _N_ IN(2,8) THEN USAGE=USAGE*1.75;
IF _N_ IN(3,9) THEN USAGE=USAGE*2;
IF _N_ IN(4,10) THEN USAGE=USAGE*.7;
IF _N_ IN(5,11) THEN USAGE=USAGE*.5;
IF _N_ IN(6,12) THEN USAGE=USAGE*.3;
END;

USAGE=ROUND(USAGE+ERR,1);

OUTPUT;
END;
RUN;

%LEVEL_SHIFTS(ATMDATA,LAST_SHIFT,ATM,DATE,USAGE);
```

Graphical Results with Seed=123

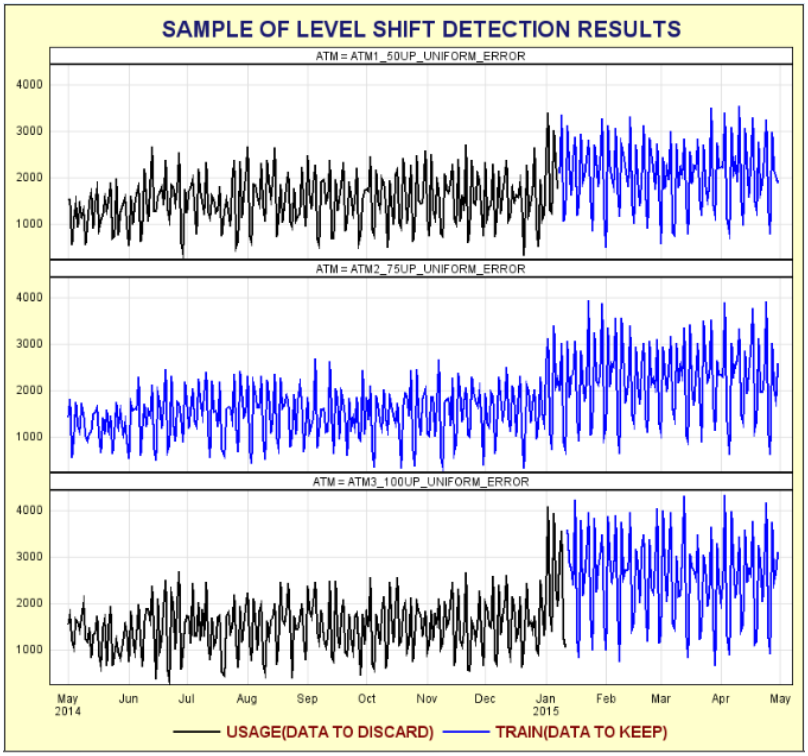

Figure 6.

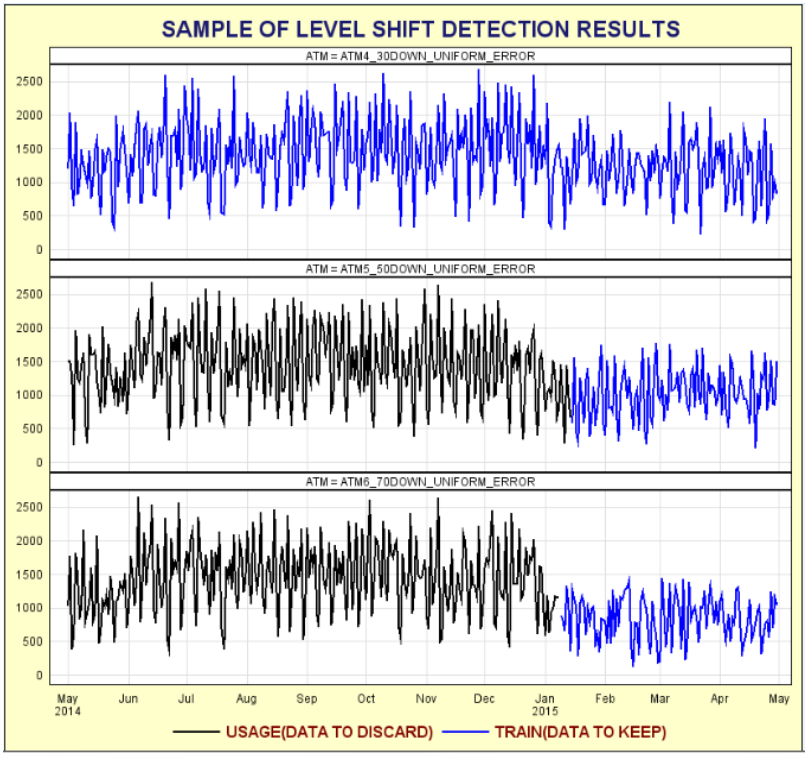

Figure 7.

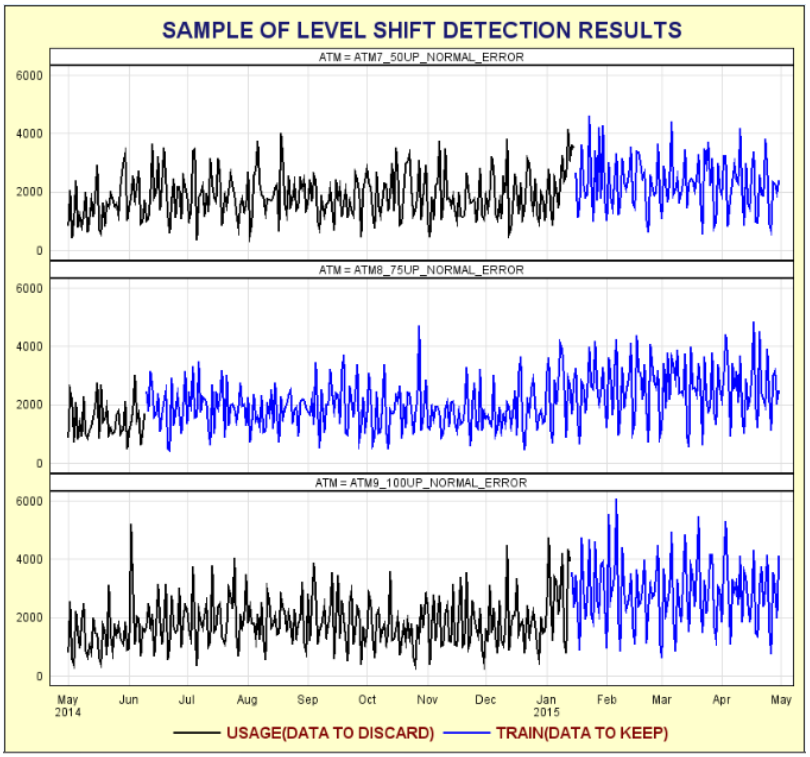

Figure 8.

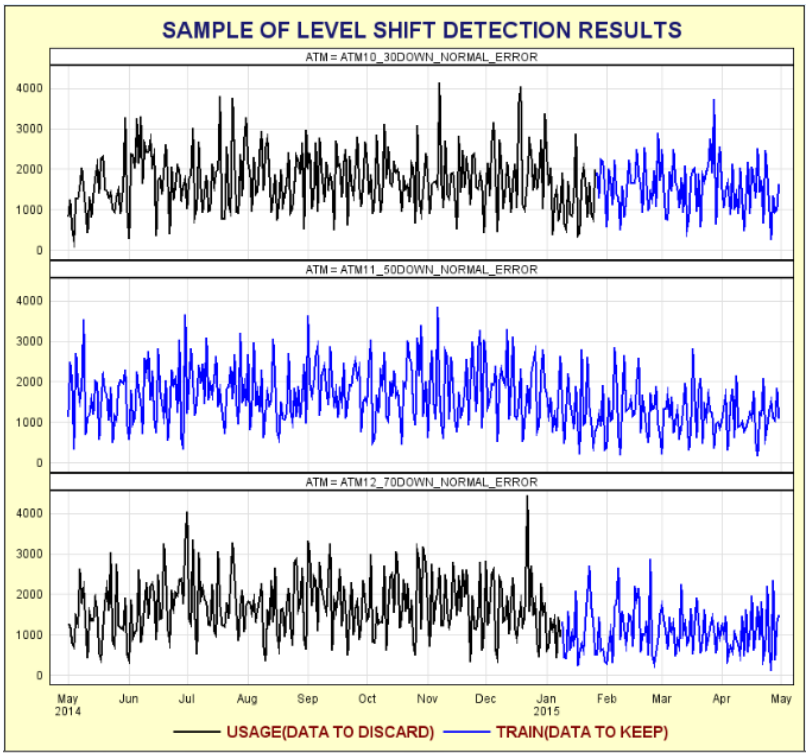

Figure 9.

Code to generate charts in Figures 6-9:

```
**********APPENDING RESULTS TO OBSERVE IN GRAPH********;
PROC SQL;*APPENDING LEVEL_DATE TO TRANCOMP FOR GRAPHING;
       CREATE TABLE SHIFTCHK AS SELECT A.*,LEVEL_DATE
              FROM ATMDATA AS A LEFT JOIN LAST_SHIFT AS B
                     ON A.ATM=B.ATM ORDER BY 1,2;
QUIT;

DATA SHIFTCHK;*CREATING A NEW VARIABLE FOR GRAPHING PURPOSES;
       SET SHIFTCHK;
       IF DATE GE LEVEL_DATE THEN
              DO;
                     TRAIN=USAGE;
                     USAGE=.;
              END;
RUN;

PROC SQL NOPRINT;*CREATING VARIABLES FOR QUICK SELECTION FOR GROUPS OF 3;
SELECT  "'"||ATM||"'" INTO:SET1 SEPARATED BY ',' FROM ATMS(OBS=3) ;
SELECT  "'"||ATM||"'" INTO:SET2 SEPARATED BY ',' FROM ATMS(FIRSTOBS=4 OBS=6) ;
SELECT  "'"||ATM||"'" INTO:SET3 SEPARATED BY ',' FROM ATMS(FIRSTOBS=7 OBS=9) ;
SELECT  "'"||ATM||"'" INTO:SET4 SEPARATED BY ',' FROM ATMS(FIRSTOBS=10 OBS=12) ;
QUIT;


*RESETTING PATH SO THE TEMPLATE CAN BE DEFINED;
ODS PATH WORK.TEMPLAT(UPDATE) SASUSER.TEMPLAT(READ) SASHELP.TMPLMST(READ);

PROC TEMPLATE;
DEFINE STATGRAPH SGDESIGN;
DYNAMIC _DATE _USAGE _TRAIN _ATM;
DYNAMIC _PANELNUMBER_;
BEGINGRAPH / BACKGROUNDCOLOR=CXFFFFCC DESIGNHEIGHT=773 DESIGNWIDTH=828;
ENTRYTITLE "SAMPLE OF LEVEL SHIFT DETECTION RESULTS"/
       TEXTATTRS=(WEIGHT=BOLD STYLE=NORMAL COLOR=MIDNIGHTBLUE SIZE=16);
LAYOUT DATAPANEL CLASSVARS=(ATM)/COLUMNS=1 PANELNUMBER=_PANELNUMBER_ CELLHEIGHTMIN=1
CELLWIDTHMIN=1
COLUMNAXISOPTS=(GRIDDISPLAY=ON DISPLAY=(TICKVALUES)) ROWAXISOPTS=(GRIDDISPLAY=ON
DISPLAY=(TICKVALUES));
       LAYOUT PROTOTYPE;
              SERIESPLOT  X=_DATE Y=_USAGE /  name="USG" lineattrs=(COLOR=BLACK  PATTERN=1
THICKNESS=2)
                     legendlabel="USAGE(DATA TO DISCARD)";
              SERIESPLOT  X=_DATE Y=_TRAIN / name="TRN" lineattrs=(COLOR=BLUE  PATTERN=1
THICKNESS=2)
                     legendlabel="TRAIN(DATA TO KEEP)" ;
       ENDLAYOUT;
       SIDEBAR / ALIGN=BOTTOM;
       DISCRETELEGEND "USG" "TRN" / ACROSS=2 OPAQUE=FALSE BORDER=FALSE
              VALUEATTRS=(WEIGHT=BOLD STYLE=NORMAL COLOR=MAROON SIZE=12) ;
    ENDSIDEBAR;
ENDLAYOUT;
ENDGRAPH;
END;
RUN;

%MACRO GRAPH_PANEL(IND);
PROC SGRENDER DATA= SHIFTCHK(WHERE=(ATM IN(&&&SET&IND.))) TEMPLATE=SGDESIGN;
DYNAMIC _DATE="DATE" _USAGE="USAGE" _TRAIN="TRAIN" _ATM="ATM";
RUN;
%MEND;
```

```
%GRAPH_PANEL(1);
%GRAPH_PANEL(2);
%GRAPH_PANEL(3);
%GRAPH_PANEL(4);
```

CONCLUSION

Determining Level Shifts is somewhat a common sense decision to make when developing a forecast model for any time-series data. Although relatively simple when done manually with charts, identifying such cutoffs are not as straight forward or 100% reliable when resorting to algorithms. There are a number of different algorithms available to help assist the forecaster with such a task- but there is a wide range of complexity and assumptions necessary to use any of them. The method presented here will hopefully be simple enough to be easily explained to broader audiences while at the same time effective enough to help improve forecast that are being produced by batch processes.

ACKNOWLEDGEMENTS

Special Thanks for their contribution to make this paper possible:
Aslam Chaudhry, Debby Feurer, James Forrest, Jason Van Hulse, Donovan Gibson, David Rodriquez, Wei Liu and Doris Grillo.

CONTACT INFORMATION

Name:Matt Bates
Company:J.P.Morgan Chase
Email:matthew.bates@chase.com

DISCLAIMER

The contents of this paper are the work of the author and do not necessarily represent the opinions, recommendations, or practices of J.P Morgan Chase.