Paper SA-06-2012

# SAS® code to compute a set of new variables from Time Series variables

**Anani K. Hoegnifioh, US Cellular, Chicago, IL**

## ABSTRACT

This study is intended to assist analysts in generating maximum number of variables using simple arithmetic operators (addition, difference, minimum, maximum, counts, and division) and one or more historical data sets. These data sets might include monthly amount paid, daily number of received customer service calls, weekly worked hours on a project, or annual total sale of a specific product. During a statistical modeling process, analysts are often confronted with the task of computing derived variables using the existing available variables. The advantage of this methodology is that the new variables may be analytically useful than the original ones. This paper provides a way to compute all the possible variables using a set of historical data such as daily, weekly, monthly, or yearly information using the simple arithmetic operators (addition, difference, minimum, maximum, counts, and division) between two or more period of an observed time-series variable and aggregate them into a derived variable. An arithmetic formula was developed to evaluate the number of possible new variables from two parameters (number of observed periods and type of operators). The code includes many SAS features that are very useful tools for SAS programmers to incorporate in their future codes such as %SYSFUNC, SQL, %INCLUDE, CALL SYMPUT, %MACRO, DICTIONARY.XXXX (where XXXX can be TABLE, COLUMN), procedures (SORT, CONTENTS), and data steps (MERGE, _NULL_) and many more. The code was originally written in SAS 9.2 but can run with any version of SAS with minor adjustment such as REMERGE option in PROC SQL.

## INTRODUCTION

Statistical data modeling process often leads to the computation of new derived variables using the existing variables. The main reason for doing so is that the derived data may constitute potentially stronger predictors for the models to be developed. This project is intended to assist modelers in generating maximum number of variables using simple arithmetic operators (addition, difference, minimum, maximum, counts, and division) and a given characteristic that can be observed historically (daily, weekly, monthly, or annually). An arithmetic formula was developed to calculate the maximum number of new variables that can be generated. The paper is divided into two different parts.
The first part contains the historical data into a modeling data format and labels the variable based on the time period. Once this section is completed then the second part will compute the new derived variables.

### Part One

Let's assume that a modeler was asked to develop a telecommunication statistical model. During the Exploratory Data Analysis, it was observed that the available data set contains a set of variables such as amount paid and number of minutes used for the last 6 months for each account in the dataset. Table 1 is an example of variables in the original dataset.

Table 1

| ID | DATE | TEST_A | TEST_B |
|---|---|---|---|
| 83382960 | 1-Oct-2010 | 21.4 | 20.4 |
| 83382960 | 1-Nov-2010 | 21.5 | 20.4 |
| 83382960 | 1-Dec-2010 | 21.7 | 20.5 |
| 83382960 | 1-Jan-2011 | 21.5 | 20.6 |
| 83382960 | 1-Feb-2011 | 21.6 | 20.4 |
| 83382960 | 1-Mar-2011 | 21.6 | 20.4 |
| 83382961 | 1-Oct-2010 | 21.8 | 20.5 |
| 83382961 | 1-Nov-2010 | 21 | 20.2 |
| 83382961 | 1-Dec-2010 | 21.5 | 20 |
| 83382961 | 1-Jan-2011 | 21.6 | 20.7 |
| 83382961 | 1-Feb-2011 | 21.4 | 20.3 |
| 83382961 | 1-Mar-2011 | 21.8 | 20.4 |

In this situation, let:

N = number of original variables (in this case, N = 2)
PERIODICVARS = repetitive variables (TEST_A and TEST_B)
NBPERIODS = number of repetitive variables (NBPERIODS = 6 months)
NONPERIODICVARS = static variables - one time variables - (ID, DATE)

The first part of the code transforms the original data set into the format shown below in table 2.

Table 2

| ID | STDATE | EDDATE | TEST_A1 | TEST_A2 | TEST_A3 | TEST_A4 | TEST_A5 | TEST_A6 | TEST_B1 | TEST_B2 | TEST_B3 | TEST_B4 | TEST_B5 | TEST_B6 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 83382960 | 1-Oct-2010 | 1-Mar-2011 | 21.5 | 21.6 | 21.7 | 22 | 22.3 | 22.3 | 21.2 | 21.8 | 21.6 | 21.9 | 22.1 | 22.2 |
| 83382961 | 1-Oct-2010 | 1-Mar-2011 | 20.3 | 20.4 | 20.5 | 20.8 | 21.1 | 21.1 | 20.2 | 20.5 | 20.3 | 20.3 | 20.4 | 21.2 |

The code includes many SAS features that are very useful tools for SAS programmers to incorporate in their future code such as %SYSFUNC, SQL, %INCLUDE, CALL SYMPUT, %MACRO, DICTIONARY.XXXX (where XXXX can be TABLE, COLUMN), SORT, CONTENTS, MERGE, MACRO _NULL_, as well as %DO … %TO … and many more. Once the dataset shown in table 2 is completed, the second part is ready to run. The code generates all the meaningful possible combinations of variables using the following arithmetic operators: Summation, Average, Difference, Minimum, Maximum, Counts, and Percentage of difference using each array variables. Table 3 shows the total number of combinations one can obtain from a set of array variables giving one set of array variables for different numbers of periods.

Table 3

| DESCRIPTIVE COMPUTATION | DESCRIPTIVE FORMULA | CONDENSED FORMULA |
|---|---|---|
| MAXIMUM NUNBER OF PERIODS (N) | N | |
| NUMBER OFCOMBINATIONS (P) | P | |
| NUMBER OF PERIODS (FROM 1 TO N) | 1, 2, 3, …, N | |
| NON MISSING VALUES FOR ALL PERIODS | 1 | 1 |
| ORIGINAL VARIABLES (INDEXED) | N | N |
| DIFFERENCE VARIABLES (POSSIBLE PAIRS) | FACT(N)/(FACT(2)*FACT(N-2)) | B |
| % DIFFERENCE VARIABLES (POSSIBLE PAIRS) | FACT(N)/(FACT(2)*FACT(N-2)) | B |
| MINIMUM VARIABLES (POSSIBLE COMBINATION) | FACT(N)/(FACT(P)*FACT(N-P)) | A |
| MAXIMUM VARIABLES (POSSIBLE COMBINATION) | FACT(N)/(FACT(P)*FACT(N-P)) | A |
| AVERAGE VARIABLES (POSSIBLE COMBINATION) | FACT(N)/(FACT(P)*FACT(N-P)) | A |
| SUMMATION VARIABLES (POSSIBLE COMBINATION) | FACT(N)/(FACT(P)*FACT(N-P)) | A |
| TOTAL NUMBER OF NEW VARIABLES = TOTAL | | 4A+2B+N+1 |

$$\boxed{TOTAL = 4\left(\frac{N!}{P!*(N-P)!}\right) + 2\left(\frac{N!}{(N-2)!*(P-2)!}\right) + N + 1}$$

**Part Two**

With our method of computation, using one characteristic observed for n periods, one can derive the total number of new variables. The next step will be to check their predictability using different data exploration techniques such as Exploratory Data Analysis, or Data Mining or any other methods to select the potential useful attributes for the further analysis. Table 4 is a summary of the potential number of variables that can be derived from an historical observation of a single variable for N consecutive periods (yearly, quarterly, monthly, weekly, daily etc…)

Table 4

| Total Number Of Derived Variables A Single Time Series Variable Observed For N Periods | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **N =** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
| **NMS(P) =** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **ORIGINAL =** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| **DIF(P) =** | 0 | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 | 55 | 66 |
| **PDF(P) =** | 0 | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 | 55 | 66 |
| **MIN(P) =** | 0 | 1 | 4 | 11 | 26 | 57 | 120 | 247 | 502 | 1013 | 2036 | 4083 |
| **MAX(P) =** | 0 | 1 | 4 | 11 | 26 | 57 | 120 | 247 | 502 | 1013 | 2036 | 4083 |
| **AVG(P) =** | 0 | 1 | 4 | 11 | 26 | 57 | 120 | 247 | 502 | 1013 | 2036 | 4083 |
| **SUM(P) =** | 0 | 1 | 4 | 11 | 26 | 57 | 120 | 247 | 502 | 1013 | 2036 | 4083 |
| **TOTAL =** | **2** | **9** | **26** | **61** | **130** | **265** | **530** | **1053** | **2090** | **4153** | **8266** | **16477** |

The section of the code that creates all the derived variables (see example in table 5) are in a separate macro and are called by the main code via a %INCLUDE statement. The code can be found in the appendix. The rest of the presentation will be a short demo of how the code works.

Table 5

| Name of Derived Variables from a period N = 3 using one Variable | | | | | Total Per Category |
|---|---|---|---|---|---|
| **NON MISSING** | NMSTEST_A | | | | 1 |
| **ORIGINAL** | TEST_A1 | TEST_A2 | TEST_A3 | | 3 |
| **DIFFERENCE** | DIFTEST_A12 | DIFTEST_A13 | DIFTEST_A23 | | 3 |
| **% DIFFERENCE** | PDFTEST_A12 | PDFTEST_A13 | PDFTEST_A23 | | 3 |
| **MINIMUM** | MINTEST_A12 | MINTEST_A13 | MINTEST_A23 | MINTEST_A123 | 4 |
| **MAXIMUM** | MAXTEST_A12 | MAXTEST_A13 | MAXTEST_A23 | MAXTEST_A123 | 4 |
| **AVGERAGE** | AVGTEST_A12 | AVGTEST_A13 | AVGTEST_A23 | AVGTEST_A123 | 4 |
| **SUMMATION** | SUMTEST_A12 | SUMTEST_A13 | SUMTEST_A23 | SUMTEST_A123 | 4 |
| | | | | | 26 |

## CONCLUSION

This method of computing the derived variables can be useful for modeling purposes and save time during the variable creation stage. The macros are very straightforward but the user needs a strong understanding of SAS programming especially in SAS MACRO in order to fully take advantage of the code.

## REFERENCES

1. Delwiche, Lora D., and Susan J. Slaughter. *The Little SAS Book: a Primer: a Programming Approach*. Cary, NC: SAS Institute, 2008.
2. *SAS 9.2 Macro Language: Reference.* Cary, NC: SAS Publishing, Feb 2009.
3. Burlew, Michele M. *SAS Macro Programming Made Easy*. Cary, NC: SAS Institute, 2006.

## ACKNOWLEDGMENTS

I would like to thank Stephen E. Brooks for his continuous assistance during the coding period.  My sincere gratitude goes also to my Manager Joel Walker and my teammates (P. Arroyo, X. Hu, J. Marek, and K. Strzalka) for their encouragement and advice.

## CONTACT INFORMATION (HEADER 1)

Your comments and questions are valued and encouraged. Contact the author at:

Anani K. Hoegnifioh
U.S. Cellular®
8410 West Bryn Mawr Ave. Suite 700
Chicago, IL 60631
Work Phone: (773) 355-3413

E-mail: Anani.hoegnifioh@uscellular.com

## APPENDIX

Source Code 1 and 2

1. First Code: “STEP 1 - VARIABLES TRANSFORMATION CODES 4 PERIODS.SAS”

```
/*****************************************************************************************/
/**** THIS DEMO CODE HAVE LIMITED NUMBER OF VARIABLES        ****/
/**** THE MAXIMUM NUMBER OF VARIABLES IS LIMITED TO 3        ****/
/**** THE MAXIMUM NUMBER OF PERIODS IS LIMITED TO 4          ****/
/**** THESE LIMITATIONS ARE DUE TO THE LIMITED NUMBER        ****/
/**** OF PAGES FOR THE PRESENTATION PAPERS.  HOWEVER         ****/
/**** ONE CAN EASILY EXPAND THEM TO ANY NUMBERS WITH         ****/
/**** SOME MINOR ADJUSTMENTS TO THE CODES.  IT IS            ****/
/**** IMPORTANT TO MENTION IT BASED ON THE EXPONENTIAL       ****/
/**** NUMBER OF THE NEW VARIABLES CAN MAKE THE PROCESS       ****/
/**** FAILED BASED ON THE HARDWARE COMPUTING CAPACITY.       ****/
/*****************************************************************************************/

/* DSNDOURCE IS THE LOCATION OR THE ORIGINAL DATA TO BE USED */
%LET DSNSOURCE = C:\SOURCEDSN;
LIBNAME SRCEDSN "&DSNSOURCE.";

%LET DSN = DSNDEMO;      /* STVAR MUST BE GREATER OR EQUAL TO 1 */
%LET STVAR = 1;          /* EDVAR MUST BE GREATER OR EQUAL TO STVAR */
%LET EDVAR = 1;          /* PERIOD MUST BE GREATER OR EQUAL TO 1*/
%LET SPERIOD = 1;
/* EPERIOD MUST BE GREATER OR EQUAL TO SPERIOD AND MUST BE LESS OR EQUAL TO 4*/
%LET EPERIOD = 4;

ODS GRAPHICS ON;
FOOTNOTE;

DATA _NULL_;
CALL SYMPUT ("NBVARS", &EDVAR. - &STVAR. + 1);
CALL SYMPUT ("NBPERIOD", %SYSFUNC(STRIP(&EPERIOD. - &SPERIOD.+ 1)));
RUN;

PROC IMPORT OUT= WORK.&DSN.0
DATAFILE = "&DSNSOURCE.\&DSN..CSV"  DBMS = CSV REPLACE;
 GETNAMES=YES;
 DATAROW=2;
RUN;
```

```
PROC SQL REMERGE NOPRINT;
CREATE TABLE &DSN.CNT AS
SELECT MONOTONIC() AS VARNB, NAME AS ORIGNAME
FROM DICTIONARY.COLUMNS
WHERE LIBNAME = "WORK"
AND MEMNAME = "&DSN.0"
AND NAME NOT IN ("ID", "DATE");

SELECT COUNT(*) AS NBORIGVARS INTO :NBORIGVARS FROM &DSN.CNT;

CREATE TABLE &DSN.PERIOD0 AS SELECT DISTINCT DATE FROM &DSN.0;

CREATE TABLE &DSN.PERIOD AS SELECT DISTINCT MONOTONIC() AS ORD, DATE FROM &DSN.PERIOD0;

CREATE TABLE &DSN.PERIOD_USED AS
SELECT DATE, MONTH(DATE) AS MONTHS, YEAR(DATE) AS YEARS
FROM &DSN.PERIOD
WHERE &SPERIOD. <= ORD <= &EPERIOD.;

SELECT COUNT(*) INTO :TOTNB_PERIOD FROM &DSN.PERIOD;

SELECT MIN(DATE) FORMAT MONYY7. AS MINDATE,
         MAX(DATE) FORMAT MONYY7. AS MAXDATE
INTO :MINDATE, :MAXDATE
FROM &DSN.PERIOD_USED;
QUIT;

%LET VARLISTS = %SYSFUNC(COMPRESS(&DSN.CNT_&NBPERIOD._PER_&NBVARS._VARS));
%LET FINALDSN = %SYSFUNC(COMPRESS(&DSN.OUT_&NBPERIOD._PER_&NBVARS._VARS));

PROC SQL REMERGE NOPRINT;
CREATE TABLE &DSN.CNT_USED AS
SELECT * FROM &DSN.CNT
WHERE &STVAR.<= VARNB <= &EDVAR.;

SELECT ORIGNAME AS VARSUSED INTO :VARUSED SEPARATED BY ", "
FROM &DSN.CNT_USED;

CREATE TABLE &DSN.USED AS
SELECT ID, DATE, &VARUSED.
FROM &DSN.0
WHERE "01&MINDATE."D <= DATE <= "01&MAXDATE."D
ORDER BY ID, DATE;

CREATE TABLE &DSN.OUT AS
SELECT DISTINCT ID,
                     MIN(DATE) FORMAT MONYY7. AS STDATE,
                     MAX(DATE) FORMAT MONYY7. AS EDDATE
FROM &DSN.USED
ORDER BY ID, STDATE;
QUIT;

%MACRO DSNPREP_0;
%DO I = 1 %TO &NBPERIOD.;
DATA _NULL_;
SET &DSN.PERIOD_USED;
IF _N_ = &I. THEN DO;
CALL SYMPUT("MONTHUSED", MONTHS);
CALL SYMPUT("YEARUSED", YEARS);
CALL SYMPUT("DATEUSED", PUT(DATE, MONYY7.));
END;
```

```
RUN;

%DO J = &STVAR. %TO &EDVAR.;
DATA &DSN.CNT_USED;
SET &DSN.CNT_USED;
IF VARNB = &J. THEN DO;
ORIGNAMES&I. = COMPRESS(ORIGNAME || &I.);
CALL SYMPUT ("ORIGNAME", ORIGNAME);
CALL SYMPUT ("ORIGNAMES", ORIGNAMES&I.);
END;
RUN;

DATA &DSN.TMP&I.&J.  (RENAME = (&ORIGNAME. = &ORIGNAMES.));
SET &DSN.USED (KEEP = ID DATE &ORIGNAME.
                              WHERE = (YEAR(DATE) = &YEARUSED.
                                             AND MONTH(DATE) = &MONTHUSED.));
RUN;

DATA &DSN.OUT;
MERGE &DSN.OUT (IN = A)
            &DSN.TMP&I.&J.(DROP = DATE);
BY ID;
IF A;
RUN;
%END;
%END;
%MEND DSNPREP_0;
%DSNPREP_0;

/***** SECOND PART OF THE CODES: CREATING NEW VARIABLES *****/
%MACRO TVAR ;
DATA &FINALDSN;
SET &DSN.OUT;
RUN;

 %INCLUDE "&DSNSOURCE.\SUBSTEP - COMPUTATIONAL MACRO 4 PERIODS.SAS";

  DATA _NULL_;
   SET &DSN.CNT_USED;
   MAXPERIOD = PUT(&NBPERIOD., 2.);
   VARFIRST = ORIGNAMES1;
   VARLAST = COMPRESS(ORIGNAME || MAXPERIOD);
   CALL SYMPUT ("VARFIRST" , VARFIRST);
   CALL SYMPUT ("VARLAST" , VARLAST);
 RUN;

 %DO NBV = 1 %TO &NBVARS.;
  DATA _NULL_;
   SET &DSN.CNT_USED;
   IF &NBV. = _N_ THEN DO;
    CALL SYMPUT ("VARNAME", STRIP(ORIGNAME));
    CALL SYMPUT ("ORIGNAME", STRIP(ORIGNAME));
   END;
  RUN;

%IF &NBPERIOD. = 1 %THEN  %DO;
  %ONEVAR;
 %END; %ELSE

%IF &NBPERIOD. = 2 %THEN  %DO;
  %TWOVARS;
  %END; %ELSE
```

```
%IF &NBPERIOD. = 3 %THEN  %DO;
  %THREEVARS;
 %END; %ELSE

%IF &NBPERIOD. = 4 %THEN  %DO;
  %FOURVARS;
 %END; %ELSE

%PUT UPDATE THE 'SUBSTEP 1 - COMPUTATIONAL MACRO 4 PERIODS.SAS'
TO ACCOMMODATE THE DESIRED VALUE OF THE MACRO VARIABLE NBPERIOD
VALUE AND ADJUST THE CODE ACCORDINGLY;

PROC SQL REMERGE NOPRINT;
CREATE TABLE &FINALDSN._CNT AS
SELECT NAME FROM DICTIONARY.COLUMNS
WHERE LIBNAME = "WORK"
AND MEMNAME = "&FINALDSN."
AND NAME NOT IN ("ID" "STDATE" "EDDATE");
QUIT;
%END;
%MEND TVAR;
%TVAR;
/**** END OF THE PROGRAM ****/
```

2. Second Code: “SUBSTEP 1 - COMPUTATIONAL MACRO 4 PERIODS.SAS”

This code is called within the first code

```
/**** START OF THE SUBSTEP CODE ****/
%MACRO ONEVAR;
 %DO A = 1 %TO &NBPERIOD.;
   DATA &FINALDSN.;
    SET &FINALDSN.;
    NMS&ORIGNAME. = &NBPERIOD. - NMISS(OF &VARFIRST. - &VARLAST.);
  RUN;
 %END;
%MEND ONEVAR;

%MACRO TWOVARS;
%ONEVAR;
 %DO A = 1 %TO &NBPERIOD.;
  %DO B = 2 %TO &NBPERIOD.;
   %IF &A. < &B. %THEN %DO;
    DATA &FINALDSN.;
     SET &FINALDSN.;
           SUM&ORIGNAME.&A.&B. = SUM(&ORIGNAME.&A., &ORIGNAME.&B.);
                MIN&ORIGNAME.&A.&B. = MIN(&ORIGNAME.&A., &ORIGNAME.&B.);
                AVG&ORIGNAME.&A.&B. = MEAN(&ORIGNAME.&A., &ORIGNAME.&B.);
                MAX&ORIGNAME.&A.&B. = MAX(&ORIGNAME.&A., &ORIGNAME.&B.);
     DIF&ORIGNAME.&A.&B. = SUM(-&ORIGNAME.&A., &ORIGNAME.&B.);
     IF &ORIGNAME.&A. NE 0 THEN PDF&ORIGNAME.&A.&B. = 100*DIF&ORIGNAME.&A.&B. / &ORIGNAME.&A.;
     ELSE PDF&ORIGNAME.&A.&B. = .;
    RUN;
   %END;
  %END;
 %END;
%MEND TWOVARS;

%MACRO THREEVARS;
%TWOVARS;
 %DO A = 1 %TO &NBPERIOD.;
```

```
  %DO B = 2 %TO &NBPERIOD.;
   %DO C = 3 %TO &NBPERIOD.;
    %IF &A. < &B. AND &B. < &C. %THEN
    %DO;
     DATA &FINALDSN.;
      SET &FINALDSN.;
      SUM&ORIGNAME.&A.&B.&C. = SUM(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C.);
                MIN&ORIGNAME.&A.&B.&C. = MIN(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C.);
                AVG&ORIGNAME.&A.&B.&C. = MEAN(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C.);
                MAX&ORIGNAME.&A.&B.&C. = MAX(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C.);
     RUN;
    %END;
   %END;
  %END;
 %END;
%MEND THREEVARS;

%MACRO FOURVARS;
%THREEVARS;
 %DO A = 1 %TO &NBPERIOD.;
  %DO B = 2 %TO &NBPERIOD.;
   %DO C = 3 %TO &NBPERIOD.;
    %DO D = 4 %TO &NBPERIOD.;
     %IF &A. < &B. AND &B. < &C. AND &C. < &D. %THEN
     %DO;
      DATA &FINALDSN.;
       SET &FINALDSN.;
      SUM&ORIGNAME.&A.&B.&C.&D. =
         SUM(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C., &ORIGNAME.&D.);
                MIN&ORIGNAME.&A.&B.&C.&D. =
         MIN(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C., &ORIGNAME.&D.);
                AVG&ORIGNAME.&A.&B.&C.&D. =
         MEAN(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C., &ORIGNAME.&D.);
                MAX&ORIGNAME.&A.&B.&C.&D. =
         MAX(&ORIGNAME.&A., &ORIGNAME.&B., &ORIGNAME.&C., &ORIGNAME.&D.);
      RUN;
     %END;
    %END;
   %END;
  %END;
 %END;
%MEND FOURVARS;
/**** END OF THE SUBSTEP CODE ****/
```

Excel Worksheet for Sample Data used:

I:\SOURCEDSN\
DSNDEMO.csv