

%RESTRUCT - SAS® macro with Proc Univariate

Milorad Stojanovic, RTI International, North Carolina

ABSTRACT

Proc Univariate has a long history as a part of the SAS Base software package. It is very handy and useful. It allows users to quickly analyze raw as well as final data at a glance. It does, however, come with a price – users often have to browse many pages of Proc Univariate report results. Macro %RESTRUCT relieves you of that burden by summarizing the data you need to examine most. The summary results are presented in a very condensed form. Users should specify their source of data, statistics they would like to have and the output file location.

KEYWORDS Macro, Proc Univariate, Proc Transpose, ODS

INTRODUCTION

When we start to talk about the usefulness of SAS Base procedures, the name Proc Univariate comes very high in rank. It allows you to get first insight with respect to the quality of your data (be it bad or good) at the beginning of data collection for each project/study/protocol. Users are able to quickly find outliers and to get some base feeling for how their data are 'breathing'. However, the procedure forces users to sift carefully through each page of output to get what they need because the listings are comprehensive and present a lot of data. The goal of developing this macro (RESTRUCT) was to relieve users of burden other than requiring them to define a few things at the beginning and provide only the output users need in a single row for each variable. Users are only asked to input values for seven macro variables.

```
%let INP_LOC =;          * Location of the input data set;
%let INP_DS =;          * Name of the input data set;
%let VAR_LIST =;        * List of variables for which you would like base statistics;
%let DS_STAT_LIST =;    * List of descriptive statistic keywords;
%let QS_STAT_LIST =;    * List of quintiles statistic keywords;
%let REPORT_LOC=;       * Location of final results;
%let NUM_OUTLIERS=;     * Number of outliers;
```

A module was added to the beginning of the macro to evaluate input parameters for characteristics that would likely cause the macro code to fail during execution. If any of the following conditions is not satisfied, error messages will be printed to the SASLOG and the macro will abort. Input parameter evaluation is a two steps process. The first step checks for the existence of input data set and for the misspelling of the statistical keywords provided. The second step performs checks on the remaining input parameters. Evaluating the input parameters first (and subsequently aborting when issues are identified) will prevent the program from executing code unnecessarily.

Input parameters checking

Step 1

- Confirm the existence of the input data set (location + data set name)
- Check for the misspelling of statistical keywords

Step 2

- Confirm that the number of variables in the input string (for later processing by PROC UNIVARIATE) is less than or equal to the pre-set value of 20.
- Confirm that the number of outliers is less than half of the total number of observations in the input data set.
- Confirm that the input string (VAR_LIST) should be free of misspellings.

If the macro input variables &INP_DS, &VAR_LIST, or &REPORT_LOC have incorrect values, the macro stops processing and prints a note in SASLOG instructing the user to correct the value of those that have been erroneously defined. Assigning &DS_STAT_LIST a value of 1 tells the macro to take, by default, all descriptive statistical keywords. Assigning a value of 0 for the macro variable &DS_STAT_LIST means the user doesn't need descriptive

statistics. If &QS_STAT_LIST is assigned value of 0 the percentile statistics are omitted. Assigning a value of 1 will initiate the processing of all percentiles (P99, P95, P90, P75, P50, P25, P10, P5, and P1).

Checking of the existence of the input data set (location + data set name).

```
%LOCAL dset dsid numobs ;
%LET dset=db.&INP_DS;          * Name of input data set ;
%let IND_LOC = 0 ;
%LET dsid=%SYSFUNC(open(&dset)); * Attempt to open input data set ;
%IF &dsid %THEN %DO ;          * If input DS exists then grab the # of observations
;
  %LET numobs = %SYSFUNC(attrn(&dsid,nobs)); * Grab the # of observations ;
  %LET rc      = %SYSFUNC(close(&dsid));
%END ;
%ELSE %DO ;
  %let IND_LOC = 1 ; * There was problem with opening input data set ;
%END ;
```

The spelling of statistical keywords is also checked in cases where the user decides to specify a specific set of them.

```
data _null_ ;
length STAT_LIST STAT_LIST1 $ 120 num_words 8 ;
STAT_LIST = "n mean std skewness uss cv sumwgt sum var kurtosis css stdmean mode range
Nmiss Nobs median Qrange max min" ;
%let IND_SP = 0 ; * Indicator shows spelling of all words from input string is correct
;
%IF "&DS_STAT_LIST" eq "1" or "&DS_STAT_LIST" eq "0" %THEN %DO ;
%END ;
%ELSE %IF "&DS_STAT_LIST" ne "1" or "&DS_STAT_LIST" ne "0" %THEN %DO ;
  STAT_LIST1 = "&DS_STAT_LIST" ;
  Num_words = COUNTW(STAT_LIST1) ;
  do I = 1 to Num_words ;
    do j = 1 to 20 ;
      if scan("&DS_STAT_LIST", i) = scan(STAT_LIST, j) then goto out ;
    end ;
    %let IND_SP = 1 ; * Indicator shows that variable name was misspelled ;
    out: ;
  end ;
%END ;
run ;
```

When conditions are not satisfied (i.e. wrong DS name or misspelled keywords) error messages are printed for the user in SASLOG and processing is aborted. For example, if a user were to specify a data set that does not exist and provided a statistical keyword string that reads "MEAN MIN **MAXX**" (where MAX is misspelled), the following messages would be printed.

```
***** -----
***** Problem with location and/or input data set name
***** Please check it and make correction
***** -----
***** -----
***** There are one or more statistical keywords misspelled
***** Please check it and make correction
***** -----
ERROR: Execution terminated by a %ABORT statement.
```

In the second input parameter evaluation step, if the user does not request specific statistics (meaning they provide DS_STAT_LIST = 0, QS_STAT_LIST = 0, and NUM_OUTLIERS = 0) the macro focuses its attention on NUM_OUTLIERS and VAR_LIST.

Below is an example of messages that would be printed to SASLOG when a problem is detected with the number of outliers or with at least one of the variables named in the variable list.

```
***** Number of outliers greater than half number of observations in input data set.
***** You should make the number of outliers smaller - to satisfy condition #outliers < Nobs / 2.
*****
***** At least one variable name from input list doesn't exist in input data set!!!
***** Please correct variable name(s) and run macro again.
*****
```

Processing of the input data set

When all input parameter conditions are satisfied, the 'real' processing of the data begins. The ultimate goal is to provide the user with a data set and reports which represent each numeric input variable (selected by the user to be processed by Proc Univariate) with a single row. A flowchart of complete processing is provided in the appendix. Below is a summary of the data processing steps:

1. Proc Contents produces the necessary variables for getting correct variable names and retrieves the number of observations in the input DS.
2. Proc Univariate produces output in the form of a SAS data set(s) for all selected numeric variables and for all requested statistical keywords and all requested percentiles.
3. All previously produced data sets are read into a new data set using the SET statement.
4. Proc Transpose converts rows to columns.
5. The initial data set (INP_DS) is processed to identify outliers.
6. Results are presented as a combination of a single SAS data set and several reports (in RTF, PDF, and HTML format).

Output from Proc Univariate is processed in a manner that restricts statistical keywords and limits corresponding processing to a single request.

First was 'preparation phase' in which all necessary variables were converted to macro variables including a number of counters (to be used later to limit DO loops or to properly positioned variables in the final report/data set).

* Converting names of numerical variables (to be processed by PROC UNIVARIATE) ;
 * to macro variable names so those would be available for later processing. ;

Begin of preparation phase

```
data _null_ ;
length VAR_LIST $ 660 ;
length ii 8 iiC $2 ;          * local index ;
VAR_LIST = "&VAR_LIST" ;
* count the number of words in a string;
* From SAS Version 9.1.3 ;
Num_Var_words = COUNTW(VAR_LIST) ;
  call symputx('Num_Var_Words', Num_Var_words); * Number of variables for processing ;
                                           * by Proc Univariate.          ;

  do ii = 1 to Num_Var_words ;
    iiC = trim(left(put(ii, 2.))) ;
    VAR_NUM = trim(left(scan(VAR_LIST, ii))) ; * Grabing one by one num variables ;
    call symputx('VAR_WORD' || iiC, VAR_NUM) ; * There are at most 20 words in the
VAR_LIST ;
  end ;
run ;
```

Preparing Statistical keywords (from string) to number of macro variables

```
data _null_ ;
length STAT_LIST $ 120 ST_var $ 8 num_words 8 ;
length ii 8 IIC $2 ;          * local index ;
%IF &DS_STAT_LIST = 1 %THEN %DO ;
  STAT_LIST = "n mean std skewness uss cv sumwgt sum var kurtosis css stdmean mode
range Nmiss Nobs median Qrange max min" ;
%END %DO ;
```

```

    %let DS_STAT_LIST1 = n mean std skewness uss cv sumwgt sum var kurtosis css stdmean
mode range Nmiss Nobs median Qrange max min ;
%END ;
%ELSE %IF "&DS_STAT_LIST" gt "1" %THEN %DO ;
    STAT_LIST = "&DS_STAT_LIST" ;
    %let DS_STAT_LIST1 = &DS_STAT_LIST ;
%END ;

* Count the number of words in a string;
* From SAS Version 9.1.3 ;
    Num_words = COUNTW(STAT_LIST) ;
    do ii = 1 to Num_words ;
        iiC = trim(left(put(ii, 2))) ;
        ST_VAR = trim(left(scan(STAT_LIST, ii)));*Grabbing one by one statistical
keywords ;
        call symputx('ST_KEYWORD' || iiC, ST_VAR) ;
    end ;
    call symputx('Num_ST_Keywords', Num_words) ;    * Get number of words in list of
statistical keywords ;
run ;

```

End of preparation phase.

```

data labs;
    set db.&INP_DS.(keep=&VAR_LIST.) ;
run ;

* Create data set with variables as rows, statistics as columns ;

proc univariate data=labs noprint;
    var &VAR_LIST ;

* Next statements create one data set per statistic and use original data set names. ;

    %IF &DS_STAT_LIST ne 0 %THEN %DO ;
        %DO ii = 1 %TO &Num_ST_Keywords ;
            %IF %upcase(&&ST_KEYWORD&ii) = N %THEN %DO ;
                output n=&VAR_LIST.          out=n;
            %END ;
            %IF %upcase(&&ST_KEYWORD&ii) = MEAN %THEN %DO ;
                output mean=&VAR_LIST.      out=mean;
            %END ;

```

Other Proc Univariate statements with statistical keywords.

Getting percentiles.

```

%IF &QS_STAT_LIST = 1 %THEN %DO ;
    output pctlpts=99 pctlpre= %DO jj=1 %TO &NUM_VAR_Words; P&jj %END ;
    out=p99 (rename=(%DO mm=1 %TO &NUM_VAR_Words; P&mm.99 = &&VAR_WORD&mm %END;)) ;
    output pctlpts=95 pctlpre= %DO jj=1 %TO &NUM_VAR_Words; P&jj %END ;
    out=p95 (rename=(%DO mm=1 %TO &NUM_VAR_Words; P&mm.95 = &&VAR_WORD&mm
%END ; )) ;

```

Other Proc Univariate statements with percentiles keywords.

```
run ;
```

Put together all requested data set in the previously determined order.

```

data db.stats;
    set
        %IF &DS_STAT_LIST = 1 %THEN %DO ;
            &DS_STAT_LIST1
        %END ;

```

```

%IF &QS_STAT_LIST = 1 %THEN %DO ;
    p99 p95 p90 p75 p50 p25 p10 p5 p1
%END ;
;
run;

```

Apply Proc Transpose to get the results in horizontal order .

```

proc transpose data=db.stats name=varname
/* Apply new variable names in same order as the concatenation of data sets in the
DATA step above. */
    out=restruct1(drop=_label_
        rename=(
            %IF &DS_STAT_LIST ne 0 %THEN %DO ;
                %DO nn = 1 %TO &Num_ST_Keywords ;
                    col&nn. = &&ST_KEYWORD&nn.
                %END ;
            %END ;

            %IF &DS_STAT_LIST = 0 and
                &QS_STAT_LIST = 1 %THEN %DO ;
                %DO pp = &QS_START %TO &QS_END ;
                    col&pp. = &&PCT&pp.
                %END ;
            %END ;

            %IF &DS_STAT_LIST ne 0 and
                &QS_STAT_LIST = 1 %THEN %DO ;
                %DO pp = &QS_START %TO &QS_END ;
                    %let rr = %EVAL(&pp - &Num_ST_Keywords) ;
                    col&pp. = &&PCT&rr.
                %END ;
            %END ;
        ));
    var &VAR_LIST ;
run;

```

The next step is to add outliers based on the value of &NUM_OUTLIERS parameter.

```

data db.restruct ;
merge restruct1 %IF &num_outliers > 0 %THEN %DO ; labs_max_min %END ; ;
run ;

```

The very last step involves presenting results. The RESTRUCT macro has two types of output:

1. Restruct SAS data set and
2. Reports (with the same content as SAS data set) in RTF, PDF and HTML format

With no any intervention user is able to get complete report (statistical keywords, percentiles, and outliers) or any combination of those three. The minimum report would be just one of those three mentioned before. It means user could get report with Statistical keywords, or Percentiles or Outliers only. An example of output in RTF format is on the next page.

Obs	varname	n	mean	std	skewness	uss	Cv	sumwgt	sum	var	kurtosis
1	wbc	120	5.763	1.2096	1.34820	4158.87	20.9902	120	691.5	1.46	1.32142
2	plt	120	203.350	41.5538	1.46284	5167626.00	20.4346	120	24402.0	1726.72	1.67942
3	creat	120	1.070	0.2955	0.82341	147.78	27.6180	120	128.4	0.09	0.01200
4	sgot	120	42.325	12.2668	-0.24042	232875.00	28.9823	120	5079.0	150.47	-1.50034
5	bili	120	0.625	0.3002	0.63604	57.60	48.0336	120	75.0	0.09	-0.57074
6	cea	118	1.293	0.8889	1.06918	289.80	68.7383	118	152.6	0.79	0.34762

Obs	css	stdmean	mode	range	Nmiss	Nobs	median	Qrange	max	min	P99	P95	P90	P75
1	174.10	0.11042	5.6	4.8	0	120	5.60	0.7	8.9	4.1	8.9	8.7	8.25	5.85
2	205479.30	3.79332	184.0	178.0	0	120	191.50	35.0	321.0	143.0	321.0	305.0	272.00	217.00
3	10.39	0.02698	0.8	1.2	0	120	1.00	0.4	1.9	0.7	1.9	1.6	1.55	1.20
4	17906.33	1.11980	25.0	36.0	0	120	44.00	25.5	59.0	23.0	59.0	58.5	57.00	53.50
5	10.73	0.02741	0.4	1.1	0	120	0.55	0.4	1.3	0.2	1.3	1.2	1.10	0.80
6	92.45	0.08183	1.3	3.5	2	120	1.10	1.2	3.6	0.1	3.6	3.6	2.60	1.80

Obs	P50	P25	P10	P5	P1	extr1	extr2	extr3	extr4	extr5	extr6	extr7	extr8
1	5.60	5.15	4.45	4.20	4.1	8.9	8.9	8.9	8.8	4.1	4.1	4.1	4.1
2	191.50	182.00	163.00	162.00	143.0	321.0	321.0	321.0	308.0	162.0	143.0	143.0	143.0
3	1.00	0.80	0.80	0.75	0.7	1.9	1.9	1.9	1.6	0.7	0.7	0.7	0.7
4	44.00	28.00	25.00	25.00	23.0	59.0	59.0	59.0	59.0	25.0	23.0	23.0	23.0
5	0.55	0.40	0.30	0.25	0.2	1.3	1.3	1.3	1.3	0.2	0.2	0.2	0.2
6	1.10	0.60	0.50	0.40	0.1	3.6	3.6	3.6	3.6	0.1	0.1	.	.

LIMITATIONS

Macro was designed to run under SAS 9.1.3 or above under Windows XP. It was not tested on other platforms. Under assumption that the largest length of macro variable is 65,534 characters macro would work with at most 1,985 numeric variables (with max name length of 32 characters) in input data set if user likes macro to do check of spelling variables in INP_VAR parameter. Also, COUNTW SAS function (from SAS 9.1.3) was used. It could be easily replaced with custom code.

CONCLUSION

With a relatively moderate level of effort, it is possible to build useful report(s) in a condensed form from the output of Proc Univariate. It is also possible to have a great deal of flexibility in presenting the results. Last but not least the macro is able to process the input data in very small amount of time – usually in the range of several seconds to one minute.

ACKNOWLEDGEMENTS

The author would like to thank Jean Lennon, Christopher Alexander, Peter Einaudi, Jean Ahn, and Robert Lee from the Education Studies Division (RTI) for all their help, comments and support in producing this paper.

REFERENCES

1. Art Carpenter, "Carpenter's Complete Guide to the SAS® Macro Language Second Edition", 2004, SAS Institute, 2004, Cary NC
2. SAS 9.2 Macro Language: Reference, 2009, Institute Inc., Cary, NC
<http://support.sas.com/documentation/cdl/en/mcrolref/61885/PDF/default/mcrolref.pdf>
3. Base SAS®9.2 Procedures Guide, 2009, SAS Institute Inc., Cary, NC
<http://support.sas.com/documentation/cdl/en/proc/61895/PDF/default/proc.pdf>

DISCLAIMER

All opinions and suggestions stated in the paper on how to restructure output from Proc Univariate do not necessarily reflect the opinions and suggestions of the Education Studies Division (RTI International).

CONTACT INFORMATION

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

Milorad Stojanovic
Education Studies Division
RTI International
3040 Cornwallis Rd
RTP, NC, 27909
Work Phone : (919) 541-7376
E-mail: milorad@rti.org

TRADEMARK INFORMATION

SAS® and all other SAS® Institute Inc. product or service names are registered trademarks or trademarks of SAS® Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies.

Appendix

**%Macro
Restruct**

