

Reducing PROC CORR Output using ODS and Data Step

Carry W. Croghan, US-EPA, Research Triangle Park, NC

ABSTRACT

The SAS procedure CORR calculates correlation coefficients, and, by default, prints the entire correlation matrix as well as descriptive statistics. A program is presented that restricts the output to the elements from above the diagonal of the correlation matrix. The default CORR output contains n^2 elements in the $n \times n$ matrix, while the reduced output contains $(n^2-n)/2$ elements. The program uses a combination of SAS's output delivery system (ODS) and data step manipulation to reduce the output. First, ODS is used to generate a SAS dataset from the correlation matrix output. Then, data step programming limits the data records and re-orient the dataset. Finally, ODS is used to output the data to Excel or other searchable file structures. The resulting output provides relevant information in an accessible and efficient format.

INTRODUCTION

The Office of Research and Development of the United States Environmental Protection Agency conducts large measurement studies. Often one of the first steps in analyzing the data generated by these studies is to look for correlations among the study measures. The SAS procedure CORR is used to calculate Pearson correlations. The output generated by PROC CORR can be unwieldy when there are a large number of measures. By default, PROC CORR prints the entire correlation matrix as well as descriptive statistics. The correlation matrix contains redundant elements: 1) the diagonal elements of the matrix are always one and 2) the matrix is symmetric. The default matrix of n^2 elements contains only $(n^2-n)/2$ elements of unique information. The presented program limits the output to just the elements above the diagonal and provides a more compact output to display the essential information. The program uses SAS's output delivery system (ODS) and data step programming.

STEP 1. USING ODS TO ACCESS THE CORRELATIONS

ODS was introduced as part of the SAS programming language with the release of version 7 and is used to manage and customize the output generated by SAS procedures. ODS facilitates the accessing of SAS-internal data tables generated by a procedure. It also allows the user to modify the output destination and structure. One approach for retrieving the names of the data tables generated by SAS is the command (ods trace on;), which generates a list, largely self-explanatory, of all of the data tables generated by a SAS procedure. PROC CORR outputs the following information to the log.

Output Added:

Name: VarInformation
Label: Variables Information
Template: base.corr.VarInfo
Path: Corr.VarInformation

Output Added:

Name: SimpleStats
Label: Simple Statistics
Template: base.corr.UniStat

Path:	Corr.SimpleStats

Output Added:	

Name:	PearsonCorr
Label:	Pearson Correlations
Template:	base.corr.StackedMatrix
Path:	Corr.PearsonCorr

The SAS-internal data tables are not automatically saved once a procedure is completed. The tables exist during the procedure run time but can be captured by ODS before deletion only by writing the data table to a dataset. This ODS command is executed prior to the procedure. The following commands create a dataset, named "p," from the SAS-internal data table containing the Pearson correlations, named "PearsonCorr."

```
ods output PearsonCorr = p ;
proc corr data=in_file pearson nosimple ;
    var x y z;
attrib _all_ label = ' ';
run;
```

The structure of the dataset is one record and one column for each variable in the var statement. This structure mimics CORR's correlation matrix form and, like CORR, stores the corresponding correlation values. Additional columns are a series of "p" variables that report the p-values of their correlation coefficients. There is also a series of "n" variables in cases where the number of observations used to calculate the correlations is not fixed across the data.

V									
a									
r									
i									
a									
0 b									
b 1					P	P	P	N	N
s e	x	y	z		x	y	z	x	y
								z	
1 x	1.00000	0.99912	0.03552	—	<.0001	0.4304	497	497	495
2 y	0.99912	1.00000	0.03705	<.0001	—	0.4094	497	500	498
3 z	0.03552	0.03705	1.00000	0.4304	0.4094	—	495	498	498

STEP 2. USING DATA STEP PROGRAMMING TO LIMIT OUTPUT.

The next step in processing the data is to limit the values to just those elements that correspond to the matrix elements that are above the diagonal. At the same time, the data are transposed.

First, to make it easier to reference the different elements, an indexed array is created for each type of variable, i.e., arrays for the correlations, sample sizes, and p-values. The (*) for the array elements allows the creation of an indexed array without having to know the exact number of elements in it. The do-loop starting point is set to “_n_+1”, which is the first element to the right of the diagonal. This do-loop's “_n_” is the internal SAS variable that always tracks the record number within the data step processing. Dim() is a SAS function that returns the dimension of the array. This function is very useful to avoid counting the number of elements in an array.

```
array ns(*) nx ny nz;  
array ps(*) px py pz;  
array vars(*) x y z;  
do i = _n_+1 to dim(ns);  
    variable1 = variable;  
    variable2 = vname(vars(i));  
    n = ns(i);  
    Pearson_corr= vars(i);  
    Pearson_pvalue = put(ps(i),PVALUE8.4);  
    output;  
end;
```

The next step is to transpose the data. The desired structure is the names of the two variables, their correlation, the p-value, and the number of observations. The matrix row element's names are saved in the variable named “VARIABLE.” The column elements are saved in separate columns and their names are the column names. The VNAME() function returns the name of the variable which is ideal for situations where the variable is part of an array. The value in that column is the correlation. The pvaluew.d format prints the p-values so they match the format used by CORR. Finally, the output statement is imbedded in the do-loop to output all the upper diagonal elements and only the upper diagonal elements.

The resulting data set follows.

Obs	variable1	variable2	n	Pearson_ corr	Pearson_ pvalue
1	x	y	497	0.99912	<.0001
2	x	z	495	0.03552	0.4304
3	y	z	498	0.03705	0.4094

STEP 3. OUTPUT THE INFORMATION USING ODS.

The final element is to output the dataset into a format that is easy to search and sort, and which may contain ancillary data. ODS permits the routing of the printout to a wide range of file structures. EXCEL was chosen for this

work in response to user preferences. Since there is no direct EXCEL output for ODS, the tagset exceltp is used to generate xml code for interpretation by EXCEL.

```
ods tagsets.excelxp file="corrs_simple.xml";  
proc print data=pearson;  
run;
```

CONCLUSION.

PROC CORR output is often large and always contains redundant information. This program outputs the essential information from CORR into a structure that is easily searched and sorted. Thus, this program dramatically reduces the amount of information that is outputted and printed. It utilizes ODS to obtain and restructure the output generated by PROC CORR, limiting the data to the elements above the diagonal by use of array processing, SAS functions; dim() and vname(), and the SAS-internal variable, _n_. The illustration is given with three variables but an expanded version of the program using macro lists is attached.

ATTACHMENT.

```
*****;  
*** Program Name:      Corrs  
*** Author:           Carry Croghan  
*** Version:          1  
***  
*** Purpose           Reduce the output from proc corr.  Output the  
***                   results to Excel.  
***  
*****;  
***   Input :   varlist list of variables  
***             in_file name of input dataset  
***             outfile name of output file that will be generated.  
***   Output:   XML file to be read into Excel  
*****;  
libname input "D:\sasdata";  
*****;  
%let varlist = x y z;  
%let outfile = corrs;  
%let in_file = input.data1;  
*****;  
***Macro creates a new list with a prefix added.  ;
```

```

%macro makelist(newlist = , addon = ,oldlist= );

%global &newlist;

%let temp = ;

%let count=1;

    %do %until(%scan(&oldlist,&count)=);

        %let listelement = %scan(&oldlist,&count);

            %let temp = &temp &addon.&listelement    ;

        %let count=%eval(&count+1);
    %end;

%let &newlist = &temp;
%mend;

%makelist(newlist = nlist, addon = n, oldlist= &varlist);
%makelist(newlist = plist, addon = p, oldlist= &varlist);

*****;

ods output  PearsonCorr = p ;

proc corr data=&in_file  pearson;
    var    &varlist;
attrib _all_ label = ' ';

run;

*****;

```

```

data Pearson(keep= variable1 variable2 n Pearson_corr Pearson_pvalue);

    set p;

    array ns(*) &nlist;
    array ps(*) &pelist;
    array vars(*) &varlist;
    do i = _n_+1 to dim(ns);
        variable1 = variable;
        variable2 = vname(vars(i));
        n = ns(i);
        Pearson_corr= vars(i);
        Pearson_pvalue = put(ps(i),PVALUE8.4);
        output;
    end;

run;

*****;
ods tagsets.excelxp file="&outfile.xml";
proc print data=pearson;
run;

*****;
**          End of program          ***;
*****;

```

ACKNOWLEDGMENTS

The United States Environmental Protection Agency through its Office of Research and Development funded and managed the research described here. It has been subjected to Agency administrative review and approved for publication. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

CONTACT INFORMATION

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

Carry Croghan

U.S. Environmental Protection Agency

MD-E205-04, 109 TW Alexander Drive

Research Triangle Park, NC 27709

Work Phone: 919 541 3184

Fax: 919 541 0905

Email: Croghan.Carry@EPA.gov

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.