Macro utility to compare multiple SAS data sets

Krish Krishnan, Statistical Programming Scientist, Quintiles

Abstract

This paper describes an efficient method to compare pairs of SAS data sets. The utility presented here will compare one or more pairs of SAS data sets and will generate a condensed user friendly report outlining the findings. The summary report generated by this utility lists above and beyond the 16 comparison results that can be obtained from the return codes stored in the automatic macro SYSINFO from PROC COMPARE.

Maximum efficiency from this macro can be achieved if the business process involves independent validation of analysis data sets, tables, listings or any other outputs that requires creation of permanent SAS data sets by both production and independent QC programmer. The macro can be modified by end user to accommodate other scenarios (e.g. to check only the data structure compliance of current data transfer to previous data transfer, print the entire PROC COMPARE output by removing the NOPRINT option etc.).

Two programs are needed and both the files containing the SAS codes are available for download through Dropbox (sign in not required):
https://www.dropbox.com/s/rwzt03u4nlh1uww/CmprDashBrd.sas
https://www.dropbox.com/s/qe65mcp9qy09d19/RunCompare.sas

Introduction

SAS Output Delivery System (ODS) is used in conjunction with traffic lighting to compare one or more pairs of data sets and a Rich Text Format (RTF) output is generated that identifies 16 comparison results that comes out of the PROC COMPARE. Traffic lighting is automatically created in the report to display in green or red whether the compare **passed** or **failed**. In addition to the 16 standard comparison results, the following additional checks are done as well:

- Will always check and flag as error, if 'production data set date is after QC data set date'.
- If enabled, the report will flag when variable ordering (position) between the production and QC data set does not match. (VARNUM option in PROC CONTENTS procedure lists the variables by the position in the data set). This check is an optional feature.
- When comparing data sets, if business process dictates any specific character (e.g. a blank space) needs to be ignored, this utility will ignore a specific character or a list of characters (optional feature).
- Specific criterion can be used (other than the PROC COMPARE default of 0.00001) to judge the equality of numeric values.

How to run the macro

Macro program is called **CmprDashbrd.sas** (Compare dash board). This macro in general is not expected to be changed unless the functionality needs to be modified. The approach taken by this paper is to only illustrate how to run this macro and how to interpret the summary report.

RunCompare.sas calls the macro CmprDashbrd and this program lists the simulated pairs of SAS data sets to compare. This program is expected to be modified to meet the end user's expectation. Both SAS files (**CmprDashbrd.sas** and **RunCompare.sas**) can be downloaded from the above mentioned Dropbox URL.

Macro CmprDashbrd can be invoked as follows:

```
%CmprDashbrd(_Dset    =%str(WORK.PREP),_Root=%str(StudyRootDirectoryGoesHere),
             _PROGDIR=%str(ProgrammingDirectory),_File=%str(RunCompare),
             _Title1 =%str(Sponsor ABCD Inc.),
             _Title2 =%str(Protocol A0000001),
             _CmpresC=%str(),_Criter =0.000000001,_VarNum =Yes) ;
```

Table 1: The following table describes the macro parameters used in CmprDashbrd macro:

	Macro parameter	Description	
1.	`_Dset`	User created SAS work data set, containing the list of production and QC data sets to compare.	
2.	`_Root`	Characters string describing the project root directory location. Used in the footnote for identification purpose only.	
3.	`_PROGDIR`	Character string describing the programming directory within the root location. Used in the footnote for identification purpose only.	
4.	`_File`	Character string providing the RTF file name. Extension (.RTF) is not required. *As a good practice, keep the same name for the SAS and RTF file.* For demonstration (in section **Example**), several RTFs are created within the same program.	
5.	`_Title1`	Title 1 to be used in the report.	
6.	`_Title2`	Title 2 to be used in the report.	
7.	`_CmpresC`	Characters that can be ignored when comparing. Impacts character variables only.	
		_CmpresC=%str()	To be used for 'as is' comparison.
		_CmpresC=%str(@)	E.g. to ignore blank space and @ sign. Note there is a space before @ sign in "**%str(@)**".
8.	`_Criter`	Criterion to be used in PROC COMPARE	
		`_Criter` =%str()	PROC COMPARE default of 0.00001 will be used.
		`_Criter=0.000000001`	Absolute relative difference of 0.000000001 is used by PROC COMPARE.
9.	`_VarNum`	Compare variable position. (VARNUM in PROC CONTENTS).	
		`_VarNum` =Yes	Compares variable position.
		`_VarNum=No`	Ignores variable position

Macro parameters **`_Root`, `_PROGDIR`, `_File`, `_Title1`, `_Title2`** are used either in the footnotes or titles to provide identifying information about the name of the SAS file, location of the SAS file and the title for the report (RTF and SAS file name are assumed to be the same). Values for macro parameters **_Dset, _Root, _PROGDIR, _File, _Title1, _Title2** will remain constant in this paper and therefore will not be repeated during the various illustrations.

How to interpret the summary report

This report will list the production SAS data set name and the production LIBREF along with the 16 comparison results (e.g. data set labels, data set types, variable labels, variable types etc.) that are obtained from the return codes in the automatic macro SYSINFO when comparing the production data set to the QC data set. In addition to these 16 results, report also lists the following two checks:

- Whether production data set date is after QC data set date.
- Whether variable position match between production and QC.

The macro will compare any pair of data sets and in the report under the column 'Outcome' will print a '**Pass**' or '**Fail**' to identify whether a compare passed or failed all the comparison checks. Column RunOrder in the report indicates the order in which the end user supplied the pairs of data sets to compare.

To interpret the 16 standard comparison results, the following macro return codes table (**Table 2**) was extracted from reference #1 (Base SAS(R) 9.2 Procedures Guide, Results: COMPARE Procedure). The footnote generated in the report will always provide a description of these 16 results.

Table 2:

Bit	Condition	Code	Hex	Description
1	DSLABEL	1	0001X	Data set labels differ
2	DSTYPE	2	0002X	Data set types differ
3	INFORMAT	4	0004X	Variable has different informat
4	FORMAT	8	0008X	Variable has different format

Bit	Condition	Code	Hex	Description
5	LENGTH	16	0010X	Variable has different length
6	LABEL	32	0020X	Variable has different label
7	BASEOBS	64	0040X	Base data set has observation not in comparison
8	COMPOBS	128	0080X	Comparison data set has observation not in base
9	BASEBY	256	0100X	Base data set has BY group not in comparison
10	COMPBY	512	0200X	Comparison data set has BY group not in base
11	BASEVAR	1024	0400X	Base data set has variable not in comparison
12	COMPVAR	2048	0800X	Comparison data set has variable not in base
13	VALUE	4096	1000X	A value comparison was unequal
14	TYPE	8192	2000X	Conflicting variable types
15	BYVAR	16384	4000X	BY variables do not match
16	ERROR	32768	8000X	Fatal error: comparison not done

When the report flags 'production data set date is > QC data set date', this indicates potential problems; a production programming update was made and QC programmer failed to implement the update and re-run the compare or production programmer ran the program and failed to communicate the information to the QC programmer. This can lead to quality issues and to ensure the report is clean and comply with the business process correctly, QC programmer will need to re-run the program and re-qc the output (this corrective measure will ensure QC data set date is after production data set date).

These 16 standard comparisons and the check to ensure 'production data set date is after QC data set date' are mandatory and will always be checked by the macro.

Following comparison checks are optional and can be enabled or disabled while invoking the macro:

- **`_VarNum` =Yes**: Whenever the variable position (or order of the variables in the data set) is critical and if this needs to be compared between the production and QC data set, this option can be enabled. When the report identifies this as an issue, production or QC programmer will need to reorder the variable. Reference # 4 provides detailed description on how to "reorder variables in a SAS data set".
- **_CmpresC:** Default for this parameter is %str(), and comparison will be done 'as is' in this case. In certain cases, production programmer can use special characters (e.g. @, |, $ etc.) while generating outputs and the corresponding image (SAS) data sets (e.g. blank space used for indenting sub categories). This may not need to be verified by the QC programmer electronically since visual cosmetic check of the production output is always expected. In this scenario, use the option **_CmpresC=**str( ) and this instructs the macro to ignore blank space when comparing production to QC data sets. Instead of a single character, list of characters can also be passed to this macro parameter. As an example, **`_CmpresC`**=%str(@ $|) will instruct the macro to ignore @, blank space, $ and |.
- **_Criter:** Default for this parameter is %str(). In this case, PROC COMPARE default criterion for judging the equality of numeric values will be used. Default as per the PROC COMPARE is 0.00001. If any other criterion needs to be used, this parameter can be used (e.g. **_Criter=**0.000000001).

For all the optional comparison checks outlined, report will dynamically change the title of the report to indicate what option is enabled.

Finally the generated report will indicate in one line for each pair of comparison data sets, whether the compare **passed** (in green) or **failed** (in red). The report will print all the failures first followed by the successful matching comparison. This is intended so the user can take corrective measure to address all failed comparison.

Examples

A section of RunCompare.SAS is presented here to support the demonstration of all examples.

```
* Simulated data sets are temporary (WORK). Utility should be used on permanent
SAS data sets ;
*+++++++++++++++++++++++++;
data class1(label="Student Data") ;
 retain Age ;
 set sashelp.class ;
if height = 69 then height = height + 0.000000001 ;
run ;
*+++++++++++++++++++++++++;
data PROD ;
 set sashelp.cars(drop=origin) ;
if Type='SUV' then Type='  SUV' ;
format type $15. ;
informat type $15. ;
label Type = 'TYPE' ;
run ;

data qc(label="Unbalanced Quotes:'(") ;
 set sashelp.cars(drop=msrp) ;
run ;
*+++++++++++++++++++++++++;
data prep(where=(order>0));
 infile datalines dsd;
 input order ProdLib :$8. QCLib :$8. ProdDset :$30. QcDset :$30. ;
datalines;
1,work,sashelp,class1,class
1,sashelp,work,class,class1
1,sashelp,sashelp,cars,cars
1,work,work,NotExist,NotExist
1,work,work,prod,qc
;
run ;
%include "CmprDashBrd.sas" ;
```

Table 3: There are five pairs of comparisons done and they are as follows:

RunOrder	Production data set (PDS)	QC data set (QDS)	Description of the comparison
1.	work.class1	sashelp.class	WORK.CLASS1 (PDS) was intentionally created after SASHELP.CLASS (QDS). When Height=69, Height was incremented by 0.000000001. Variable position for AGE has been changed.
2.	sashelp.class	work.class1	Role of (PDS) and (QDS) reversed.
3.	sashelp.cars	sashelp.cars	(PDS) and (QDS) are same. Clean compare is expected.
4.	work.NotExist	work.NotExist	Comparing non existing data sets.
5.	work.prod	work.qc	In addition to several standard compare differences, TYPE='SUV' is different from TYPE=' SUV'.
*Note: **RunOrder** will be referred frequently in the examples.*			

In all the examples listed below note that the macro parameters _Dset, _Root, _PROGDIR, _File, _Title1, _Title2 are constant. Section "How to run the macro?" provides the default values for these macro parameters. Refer to Table 3, whenever RunOrder is used to identify the pairs of data sets compared. QC data set name is not printed in the report.

Example 1:

Let us say macro is invoked as follows:

```
%CmprDashbrd(_CmpresC=%str(),
             _Criter =%str(),
             _VarNum=Yes
              ) ;
```

Since _CmpresC=%str(), comparison will be done 'as is'. Since _Criter=%str(), default PROC COMPARE Criterion value of 0.00001 will be used.

- RunOrder=1: VALUE, PROD_GT and VARNUM are identified as compare differences.
- RunOrder=2: VALUE and VARNUM are identified as compare differences (and not PROD_GT).
- RunOrder=3: Comparison is clean.
- RunOrder=4: Nonexistent data set is flagged as ERROR.
- RunOrder=5: As expected PROD has various compare difference when compared to QC.

See **Output 1** for the report displayed by using this macro.

Example 2:

```
%CmprDashbrd(_File   =%str(RunCompare2),
             _CmpresC=%str(),
             _Criter =%str(0.000000001),
             _VarNum=Yes
              ) ;
```

Compare results using RunOrder has the identifier:

- Since _Criter =0.000000001, RunOrder=1 and 2 does not have VALUE difference any more.
- As expected, all others are similar to Example 1.

See **Output 2** for the report displayed by using this macro.

Example 3:

```
%CmprDashbrd(_File   =%str(RunCompare3),
             _CmpresC=%str( ),
             _Criter =%str(),
             _VarNum=Yes
              ) ;
```

Since _CmpresC=%str( ), spacing is ignored during comparison and hence RunOrder=5 no longer has VALUE difference. All others are similar to Example 1.

See **Output 3** for the report displayed by using this macro.

Example 4:

```
%CmprDashbrd(_File    =%str(RunCompare4),
             _CmpresC=%str( ),
             _Criter =%str(0.000000001),
             _VarNum=No
              ) ;
```

In this example, spacing differences are ignored for character variables and absolute relative difference of 0.000000001 is used by PROC COMPARE and variable ordering is ignored. This example is also compared against Example 1.

- RunOrder=1: Small increment made to Height and ordering (for variable AGE) is ignored.
- RunOrder=2: Clean compare. Same reasons as RunOrder=1 and also 'QDS date is > PDS date'.
- RunOrder=3: As always compare matched 100%.
- RunOrder=4: Remains the same as Example 1.
- RunOrder=5: VALUE difference of Type='SUV' versus Type=' SUV' is no longer present since spacing is ignored.

See **Output 4** for the report displayed by using this macro.

Conclusion

This utility is a handy tool for QC programmer to efficiently find out quickly and easily if there are any compare differences. This can be used by the lead QC programmer to ensure all output comparisons matched electronically. This report can be used as an audit trail to indicate all outputs passed independent electronic validation before the deliverable is released. Last but not the least, this tool has been a huge timesaver whenever all study outputs have to re-run due to underlying datasets getting changed during the deliverable or when several re-runs are planned or expected during the course of the project.

References

1. Base SAS(R) 9.2 Procedures Guide, Results: COMPARE Procedure
   http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm#a000146743.htm
2. Maria Y. Reiss (2003). DARE TO COMPARE, Tailoring PROC COMPARE Output.
   http://analytics.ncsu.edu/sesug/2003/TU01-Reiss.pdf
3. Kavitha Madduri (2012). PharmaSUG 2012 - Paper AD24, Let's compare two SAS® libraries!
   http://www.pharmasug.org/proceedings/2012/AD/PharmaSUG-2012-AD24.pdf
4. Imelda C (2002). SESUG 2002 - Reordering Variables in a SAS Data Set
   http://analytics.ncsu.edu/sesug/2002/PS12.pdf

Acknowledgments

Author would like to thank the employer Quintiles for the approval to present this paper at this SCSUG forum and thanks to SCSUG 2013 board members Kenny Bissett and Lizette Alonzo for accepting the abstract and paper. Special thanks to Tommy Retzlaff and Prema Ranjit for their valuable feedback on this paper.

Contact information

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

Krish Krishnan
Statistical Programming Scientist
Quintiles
Krish.Krishnan@quintiles.com
Krish.Krishnan@usa.com

Output 1:

Compare is done 'as is' when comparing character variables. Macro variable VARNUM=Yes.
Default CRITERION (0.00001) is used in 'PROC COMPARE'

Obs	OutCome	RunOrder	PROD Libref	PROD Dset Name	DSLABEL	DSTYPE	INFORMAT	FORMAT	LENGTH	LABEL	BASEOBS	COMPOBS	BASEBY	COMPBY	BASEVAR	COMPVAR	VALUE	TYPE	BYVAR	ERROR	PROD_GT	VARNUM
1	Fail	1	work	class1													X				X	X
2	Fail	2	sashelp	class													X					X
3	Fail	4	work	NotExist																X		
4	Fail	5	work	prod	X		X	X		X					X	X	X					X
5	Pass	3	sashelp	cars																		

Output 2:

Compare is done 'as is' when comparing character variables. Macro variable VARNUM=Yes.
CRITERION=0.000000001 option is used in 'PROC COMPARE'

Obs	OutCome	RunOrder	PROD Libref	PROD Dset Name	DSLABEL	DSTYPE	INFORMAT	FORMAT	LENGTH	LABEL	BASEOBS	COMPOBS	BASEBY	COMPBY	BASEVAR	COMPVAR	VALUE	TYPE	BYVAR	ERROR	PROD_GT	VARNUM
1	Fail	1	work	class1																	X	X
2	Fail	2	sashelp	class																		X
3	Fail	4	work	NotExist																X		
4	Fail	5	work	prod	X		X	X		X					X	X	X					X
5	Pass	3	sashelp	cars																		

Output 3:

Compare will ignore characters within left and right arrow -> <- when comparing. Macro variable VARNUM=Yes.
Default CRITERION (0.00001) is used in 'PROC COMPARE'

Obs	OutCome	RunOrder	PROD Libref	PROD Dset Name	DSLABEL	DSTYPE	INFORMAT	FORMAT	LENGTH	LABEL	BASEOBS	COMPOBS	BASEBY	COMPBY	BASEVAR	COMPVAR	VALUE	TYPE	BYVAR	ERROR	PROD_GT	VARNUM
1	Fail	1	work	class1													X				X	X
2	Fail	2	sashelp	class													X					X
3	Fail	4	work	NotExist																X		
4	Fail	5	work	prod	X		X	X		X					X	X						X
5	Pass	3	sashelp	cars																		

Output 4:

Compare will ignore characters within left and right arrow -> <- when comparing. Macro variable VARNUM=No.
CRITERION=0.000000001 option is used in 'PROC COMPARE'

Obs	OutCome	RunOrder	PROD Libref	PROD Dset Name	DSLABEL	DSTYPE	INFORMAT	FORMAT	LENGTH	LABEL	BASEOBS	COMPOBS	BASEBY	COMPBY	BASEVAR	COMPVAR	VALUE	TYPE	BYVAR	ERROR	PROD_GT	VARNUM
1	Fail	1	work	class1																	X	NA
2	Fail	4	work	NotExist																X		
3	Fail	5	work	prod	X		X	X		X					X	X						NA
4	Pass	2	sashelp	class																		NA
5	Pass	3	sashelp	cars																		NA