

What Do You Mean My CSV Doesn't Match My SAS® Dataset?

Patricia Guldin, Merck & Co., Inc; Young Zhuge, Merck & Co., Inc.

ABSTRACT

Statistical programmers are responsible for delivering high quality and reproducible analysis data sets to statisticians, modelers and other quantitative scientists. Regardless of the format (e.g. SAS data set or .csv), the content of the data sets should be identical. Converting SAS data sets to other formats can be easily accomplished in SAS, but the consistency between the output files must be included in the quality control checks. An example will be given where tables and figures created by a statistician (using the SAS data set) and those created by a modeler (using the .csv data set) were different. We will provide the results of our exploration into why the inconsistencies occurred and the steps taken to ensure reliability for subsequent data exports / format conversions.

INTRODUCTION

Providing programming support to Quantitative Pharmacology and Pharmacometrics requires delivering analysis data sets in both SAS data set and .csv formats. Modelers need the .csv as inputs to some of the applications they use. Exporting or converting SAS data sets to .csv and other formats is quite straightforward and common but should not be done without thought. During an independent quality control check we discovered that the counts in the tables created by the statistician, who used the SAS data set, and the modeler, who used the .csv, did not match as expected. The root cause was identified as the data sets and .csv not having the same number of records. This paper will explore why the difference occurred and the steps taken to identify and correct this situation and assure customers that our deliverable quality is intact.

BACKGROUND – WHAT HAPPENED AND WHY

Our Statistical Programming group recently expanded support to additional areas including Quantitative Pharmacology and Pharmacometrics, PKPD and Modeling and Simulation. In supporting this group, our team often receives data from sources outside our central database. For example, ad-hoc estimates of PK parameters, i.e. AUC can sometimes be provided by vendors and are received as excel or .csv formats, and the PK assay results are in a system that works well with the workbench used by the modelers but is not yet compatible with the central clinical data repository. This data is all converted to SAS data sets and used to produce deliverables. In the particular situation described in this paper, virologic resistance variables were derived by clinical scientists and statisticians based on lab results, provided to a programmer in excel, and used to create an input SAS data set that was used by the PKPD programmers to produce an analysis data set.

One issue with using data from other sources is that these formats can introduce non printable special characters into character variables in the data. When converting excel or .csv files to SAS data sets there are no warnings to alert the programmer that these special characters are present and because these are non printable they may go undetected. Non printable special characters may or may not affect the data. Some cause a carriage return which can be found by comparing the number of records, but others can cause the meaning of a value to be different. The paper "Non Printable & Special Characters: Problems and how to overcome them", listed in the references section, does a nice job of explaining special characters and how to identify and remove or replace them. It also gives an example of how special characters can change the form or meaning of a character value. One of these special non printable characters was present in the source excel file that produced the virologic data set which in turn was used to create the PKPD data set and .csv where the difference in counts was seen. The special character had the effect of inserting a carriage return in the middle of the data set record. When the data set was converted to .csv, the return forced the remainder of the record to a new record. The result was that the one record in the SAS data set became 2 records in the .csv with missing values for some variables in each record.

%MKCSV

Once this was discovered we realized that we needed to be more aware of special characters and consult with the project teams on how to deal with them if they are found. Our customers want assurance that our deliverables, regardless of format are consistent and dependable. We need a standard way to handle special characters and assure our customers. We use a macro (%mkcsv) which was developed to convert SAS data sets to .csv files. Note there are multiple ways to convert SAS to .csv. We use SAS DATA step and FILE statements to convert SAS data files to .csv files. This code can ensure that missing values of numeric variables are "." in the .csv file which is required by NONMEM (Non-linear mixed effects modeling). We do not use SAS PROC EXPORT, because missing values for numeric variables are shown as blank in the resulting .csv files. Commas are also replaced with spaces in %mkcsv. We used criterion=.001 to ignore the small differences in numeric values due to the format change during

the conversion to .csv and back to a SAS data set. The criterion can be adjusted as appropriate for the data.

To provide a standard and to have documented confidence in reliability regardless of format, we decided to update %mkcsv. The basics of the update are converting the .csv back to a SAS data set and then comparing it with the original data set. If differences are found output a message in the log. Logs are saved and checked for all deliverables so the log serves as documentation for the customer that the formats match. No messages indicate that the .csv file and SAS data set are the same. The macro should check that the record count is the same between the two files to catch special non printable characters that inserted a carriage return. It also should check for special characters that change the meaning of a value, even though we have not experienced this situation to date. Special characters are only expected in character variables, not in numeric variable values.

The first check is for special characters that act as carriage returns. These cause additional rows to be output in the .csv file so a comparison of the number of records in each will identify that there is an issue. A message is output to the log including the first observation that is in error.

The second check is for special characters that change the meaning of a value. These special characters do not cause the number of records to differ, but as described in the papers listed in the Reference section below, can cause differences in counts. With PROC COMPARE we can output any differences to a data set. If the difference data set exists a message is output to the log and the data set can be used to identify where the issue is.

SYSINFO

While doing research for this paper we learned that PROC COMPARE stores a return code in the automatic macro variable SYSINFO. To get the value of SYSINFO you must execute a PROC COMPARE and you must retrieve the value of SYSINFO immediately after the PROC COMPARE because SYSINFO is reset with the start of any SAS step. The value of this the return code provides information about the result of the comparison. This return code is the sum of the codes for the conditions that are true and since the codes are ordered and scaled, the value of SYSINFO can provide a sense of how different the two files are. For example if SYSINFO is less than 64, the differences are in labels and formats only. If SYSINFO is 64 or greater there are more severe differences such as observations in one data set and not the other. If SYSINFO is 4096 or greater there are value differences. Displaying the value of SYSINFO was helpful in validation of the code used to check for differences due to special characters so we included it to provide more information about the results of the PROC COMPARE. The full table of macro return codes is available in Table 1 below.

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
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

Table 1 Macro Return Codes

EXAMPLES

EXAMPLE 1

Example 1 has no special characters and no value differences. In this example it is expected that the .csv file matches the original SAS data set. The message in the log (Display 1) and the PROC COMPARE (Display 2) both confirm the same. Note the return code is 60, indicating only format differences.

OK! CSV data is same as input dataset devtest.example1

NOTE: There were 1 observations read from the data set WORK.DIF.
 NOTE: The data set WORK.DIF_ has 1 observations and 5 variables.
 NOTE: DATA statement used (Total process time):
 real time 0.01 seconds
 cpu time 0.01 seconds

"return code = " 60

Display 1 Example 1 SAS Log – no differences

Comparison of DEVTEST.EXAMPLE1 with WORK.X_C (Method=RELATIVE(2.22E-11), Criterion=0.001)					
Listing of Common Variables with Differing Attributes					
Variable	Dataset	Type	Length	Format	Informat
SAFFL	DEVTEST.EXAMPLE1	Char	1		
	WORK.X_C	Char	1	\$1.	\$1.
ETIME	DEVTEST.EXAMPLE1	Num	8		
	WORK.X_C	Num	8	BEST12.	BEST32.

Observation Summary		
Observation	Base	Compare
First Obs	1	1
Last Obs	218	218

Number of Observations in Common: 218.
 Total Number of Observations Read from DEVTEST.EXAMPLE1: 218.
 Total Number of Observations Read from WORK.X_C: 218.

Number of Observations with Some Compared Variables Unequal: 0.
 Number of Observations with All Compared Variables Equal: 218.

Values Comparison Summary		
Number of Variables Compared with All Observations Equal:	55.	
Number of Variables Compared with Some Observations Unequal:	0.	
Total Number of Values which Compare Unequal:	0.	
Total Number of Values not EXACTLY Equal:	1856.	
Maximum Difference Criterion Value:	0.00026108.	

Display 2 Example 1 SAS PROC COMPARE – format only differences

EXAMPLE 2

Example 2 is the situation that prompted this paper. There was a non printable special character in the variable VARB3 for UID 61191554 that caused a carriage return. The value for VARB5 for UID 61191554 was pushed to the

next record into the AN variable. The result was two rows with missing values in each of them. In this example we expect to see a difference in the count of records and a message in the log that the .csv file does not match the original SAS data set. Display 3 shows the original SAS data set and the resulting .csv with the data pushed to the next row. Display 4 shows the message in the log indicating that there is a difference at observation 54 for UID=61191554. The return code of 12476 is another indicator that there are values differences, which can be seen in Display 5.

UID	AGE	VARV3	VARV5	VARB3	VARB5
61191549	36				
61191550	43	n/a	n/a	WT	WT
61191551	52				
61191552	50				
61191553	51				
61191554	35	Y56F V107I/V	WT	WT	WT
61191555	44				
61191556	35				
61191557	53				

	A	B	C	D	E	F
1	UID	AGE	VARV3	VARV5	VARB3	VARB5
50	61191549	36				
51	61191550	43	n/a	n/a	WT	WT
52	61191551	52				
53	61191552	50				
54	61191553	51				
55	61191554	35	Y56F V107I/V	WT	WT	
56		WT				
57	61191555	44				
58	61191556	35				
59	61191557	53				

Display 3 Example 2 original SAS data set and .csv file with carriage return

```
ERROR: Difference found between original dataset devtest.example2a and CSV data. Check the CSV
obs=      54, uid=      61191554 to determine which variable has a problem.
ERROR: Check the original dataset devtest.example2a, obs=      54 uid=      61191554 to
determine which variable has a problem.

NOTE: There were 1 observations read from the data set WORK.DIF.
NOTE: The data set WORK.DIF_ has 1 observations and 5 variables.
NOTE: DATA statement used (Total process time):
      real time          0.01 seconds
      cpu time           0.01 seconds

"return code = " 12476
```

Display 4 Example 2 SAS Log - differences

The COMPARE Procedure				
Comparison of DEVTEST.EXAMPLE2A with WORK.X_C				
(Method=RELATIVE(2.22E-11), Criterion=0.001)				
Data Set Summary				
Dataset	Created	Modified	NVar	NObs
DEVTEST.EXAMPLE2A	15JUN16:12:05:58	15JUN16:12:05:58	7	218
WORK.X_C	15JUN16:12:05:59	15JUN16:12:05:59	7	219

Observation Summary		
Observation	Base	Compare
First Obs	1	1
First Unequal	54	54
Last Unequal	218	218
Last Match	218	218
Last Obs	.	219

Number of Observations in Common: 218.
 Number of Observations in WORK.X_C but not in DEVTEST.EXAMPLE2A: 1.
 Total Number of Observations Read from DEVTEST.EXAMPLE2A: 218.
 Total Number of Observations Read from WORK.X_C: 219.

The COMPARE Procedure					
Comparison of DEVTEST.EXAMPLE2A with WORK.X_C					
(Method=RELATIVE(2.22E-11), Criterion=0.001)					
Value Comparison Results for Variables					
Unique Subject ID					
	Base	Compare			
Obs	UID	UID	Diff.	% Diff	
55	61191555	.	.	.	.

Display 5 Example 2 SAS PROC COMPARE output – differences

EXAMPLE 3

Example 3 is the situation where there are special characters that change the meaning of a value but do not change the count of records. We have not experienced this issue so we mocked it by adding commas for demonstration purposes and to show how the %mkcsv would handle this situation. In this example we do not expect to see a difference in the count of records but we do expect to see a message in the log that the .csv file does not match the original SAS data set. Display 6 shows the original SAS data set with commas in the values and Display 7 shows the same data set after commas are removed by %mkcsv but before the conversion to a .csv file. Display 8 shows the resulting .csv and Display 9 shows the result of converting the .csv file back to a SAS data set. The data sets in Displays 7 and 9 are compared to see if the .csv matches the original SAS data set. Because space Y does not equal Y to a computer, the .csv file and original data set do not match and we see a message in the log in Display 10 indicating that there is a difference at observations 3 and 5. In Displays 7 and 11 you can see that space Y does not equal Y. The return code of 4140 is another indicator that there are values differences, which can be seen in Display 11.

	SAFFL	UID	AN	AGE	BMI	SEXF
1	Y	35151451	151451	44	28.72	0
2	Y	35151452	151452	63	27.55	1
3	.Y	35151453	151453	43	21.22	1
4	Y	35151454	151454	31	27.08	0
5	.Y	35151455	151455	37	25.70	1
6	Y.Y	35151456	151456	35	24.86	1
7	Y.Y	35151457	151457	24	25.18	0
8	Y	35151458	151458	57	27.18	0

Display 6 Example 3 original SAS data set

	SAFFL	UID	AN	AGE	BMI	SEXF	ord	obs	uid_b
1	Y	35151451	151451	44	28.72	0	1	1	35151451
2	Y	35151452	151452	63	27.55	1	2	2	35151452
3	Y	35151453	151453	43	21.22	1	3	3	35151453
4	Y	35151454	151454	31	27.08	0	4	4	35151454
5	Y	35151455	151455	37	25.70	1	5	5	35151455
6	YY	35151456	151456	35	24.86	1	6	6	35151456
7	YY	35151457	151457	24	25.18	0	7	7	35151457
8	Y	35151458	151458	57	27.18	0	8	8	35151458

Display 7 Example 3 original SAS data set with commas removed by %mkcsv

	A	B	C	D	E	F
1	SAFFL	UID	AN	AGE	BMI	SEXF
2	Y	35151451	151451	44	28.72	0
3	Y	35151452	151452	63	27.55	1
4	Y	35151453	151453	43	21.22	1
5	Y	35151454	151454	31	27.08	0
6	Y	35151455	151455	37	25.7	1
7	YY	35151456	151456	35	24.86	1
8	YY	35151457	151457	24	25.18	0
9	Y	35151458	151458	57	27.18	0
10						

Display 8 Example 3 .csv file

	SAFFL	UID	AN	AGE	BMI	SEXF	ord	obs	uid_c
1	Y	35151451	151451	44	28.72	0	1	1	35151451
2	Y	35151452	151452	63	27.55	1	2	2	35151452
3	Y	35151453	151453	43	21.22	1	3	3	35151453
4	Y	35151454	151454	31	27.08	0	4	4	35151454
5	Y	35151455	151455	37	25.7	1	5	5	35151455
6	YY	35151456	151456	35	24.86	1	6	6	35151456
7	YY	35151457	151457	24	25.18	0	7	7	35151457
8	Y	35151458	151458	57	27.18	0	8	8	35151458

Display 9 Example 3 .csv file converted back to SAS data set for comparison

There are 2 differences between csv and input dataset devtest.example3b. These observations have differences: 3, 5

NOTE: There were 1 observations read from the data set WORK.DIF.
 NOTE: The data set WORK.DIF_ has 1 observations and 5 variables.
 NOTE: DATA statement used (Total process time):
 real time 0.00 seconds
 cpu time 0.00 seconds

"return code = " 4140

Display 10 Example 3 SAS log - differences

Variables with Unequal Values

Variable	Type	Len	Label	Ndif	MaxCrit
SAFFL	CHAR	3	Safety Population Flag	2	

Value Comparison Results for Variables

		Safety Population Flag	
	Obs	Base Value SAFFL	Compare Value SAFFL
	3	Y	Y
	5	Y	Y

Display 11 Example 3 PROC COMPARE - differences

%MKCSV CODE

Here are pieces of the code for %mkcsv with commenting added to describe them:

```
%MACRO mkcsv(indt=, inuid=, outdt=, outdir=);

/*definition of macro variables: indt= the original SAS data set; inuid= unique
identifier, i.e. a subject number; outdt= the .csv file name, usually matches
the data set name without the libname; outdir= the directory where the .csv is
output*/

...

/*remove commas from character variables*/...

/*get variable names to use as column headers in csv*/...

/*create csv from SAS data set*/...

/*convert the newly created csv back to a SAS data set*/...

/*Compare the 2 SAS data sets and output differences to work.result*/
PROC COMPARE BASE=&indt COMPARE=x_c criterion=.001
OUT=result outnoequal;
RUN;

/*store the value of SYSINFO immediately after PROC COMPARE*/
%LET permcode=&sysinfo.;

/*store the count of observations in work.result in nob macro variable*/
PROC SQL NOPRINT;
SELECT COUNT(*) INTO : nob
FROM WORK.result;
QUIT;
```

```

/*initialize to . for cases where there are no differences to output then
store the observation numbers of the records with differences from
work.result*/
%LET varlist = .; PROC SQL NOPRINT;
SELECT _obs_ INTO :varlist separated by ', '
FROM result;
QUIT;

/*check the record counts between the original data set(x_b)and the csv
converted to a data set(x_c)*/
DATA &indt; SET &indt; ord+1; obs=ord; uid_b=&inuid; RUN;
DATA x_c; SET x_c; ord+1; obs=ord; uid_c=&inuid; RUN;

/*store the record counts in macro variables*/...
/*identify the observation where differences in the record count starts*/
DATA find(KEEP=obs);
MERGE &indt(KEEP=obs &inuid uid_b in=in1) x_c(KEEP=obs &inuid uid_c in=in2);
by obs;
IF in1 and in2 and uid_b^=uid_c & uid_c=.;
OUTPUT;
obs=obs-1; OUTPUT;
RUN;
PROC SORT; by obs; RUN;

DATA find1; MERGE x_c(in=in1) find(in=in2);
by obs;
IF in1 & in2;
RUN;

%LET uid_=' '; %LET obs_=' ';
PROC SQL NOPRINT;
SELECT &inuid INTO :uid_ FROM find1(where=(&inuid>));
SELECT obs INTO :obs_ FROM find1(where=(&inuid>));
QUIT;

/*assign the appropriate log message text to macro variables*/
DATA dif;
dif=.;
FORMAT difc difc2 $300.;
cc=ABS(INPUT(&totnum_c,best.));
bb=ABS(INPUT(&totnum_b,best.));
dif=ABS(bb-cc);

```

```

/*record counts are equal */
IF dif=0 THEN do;
  difc="OK! CSV data is same as input data set &indt.";
  difc2=' ';
END;

/*record counts are equal, result has observations but they are label and
format issues only*/
IF dif=0 and &nobs>0 and &permcode < 4096 THEN do;
  difc="OK! CSV data is same as input data set &indt.";
  difc2=' ';
END;

/*record counts are equal, result has observations that are value differences*/
IF dif=0 and &nobs>0 and &permcode GE 4096 THEN do;
  difc="There are &nobs. differences between csv and input data set &indt.. These
observations have differences: &varlist";
  difc2=' ';
END;

/*record counts are not equal */
ELSE IF dif>=1 THEN do;
  difc="ERROR: Difference found between original data set &indt. and CSV data.
Check the CSV obs=&obs_, uid=&uid_ to determine which variable has a problem.";
  difc2="ERROR: Check the original data set &indt, obs=&obs_ &inuid=&uid_ to
determine which variable has a problem.";
END;

RUN;

/*put the message in the log*/
PROC SQL NOPRINT;
SELECT difc INTO :difc_ FROM dif;
SELECT difc2 INTO :difc2_ FROM dif;
QUIT;

DATA dit_; SET dif;
%PUT &difc_;
IF dif GE 1 THEN do; %PUT &difc2_; END;
RUN;

/*put the SYSINFO return code from PROC COMPARE in the log*/
%PUT "return code = " &permcode;
%MEND mkcsv;

```

CONCLUSION

Programmers are responsible for delivering high quality and reproducible outputs. It is important that quality control checks are done even for common tasks, such as converting SAS data sets to other formats, because our customers rely on us for quality. When the unexpected happens it forces us to improve but we should look for ways in our daily work to assure that our deliverable quality is intact.

REFERENCES

- Sridhar R Dodlapati, Praveen Lakkaraju, Naresh Tulluru and Zemin Zeng. 2010. "Non Printable & Special Characters: Problems and how to overcome them". Proceedings of the PharmaSUG 2010 Conference. Available at <http://www.lexjansen.com/pharmasug/2010/CC/CC13.pdf>
- Bob Hull, Robert Howard. 2013. "Useful Tips for Handling and Creating Special Characters in SAS®". Proceedings of the PharmaSUG 2013 Conference Available at <http://www.pharmasug.org/proceedings/2013/CC/PharmaSUG-2013-CC30.pdf>
- SAS online documentation

ACKNOWLEDGMENTS

We would like to acknowledge our manager, Jing Su, and the rest of the PKPD programming team for their support and contributions to the research and solution discussed in this paper.

CONTACT INFORMATION

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

Name: Patricia Guldin
Enterprise: Merck & Co., Inc.
Address: 351 North Summneytown Pike
City, State ZIP: North Wales, PA 19454
E-mail: patricia.guldin@merck.com

Name: Young Zhuge
Enterprise: Merck & Co., Inc.
Address: 351 North Summneytown Pike
City, State ZIP: North Wales, PA 19454
E-mail: young.zhuge@merck.com

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.