PhUSE 2016

Paper PP18

Fast and high quality validation process

Alessia Sacco, Valos, Genova, Italy
Dmitri Petratchenko, Valos, Genova, Italy
Andrea Parodi, Valos, Genova, Italy

ABSTRACT

Double programming is a must in validating databases, tables and listings. However, once these objects are produced twice, how to compare them in an efficient and free of oversight way? This paper describes our process of validation that ensure a fast and high quality validation process.

INTRODUCTION

The purpose of a validation plan is to describe the validation activities performed for checking errors in databases, tables, listings and figures produced. All the objects (datasets and outputs) that require validation have to be listed in it. Three levels of validation exist:

- Most Critical (MC): for datasets and outputs deemed "most critical", double programming by an independent (i.e. the owner of the object and the validator of the object are different persons) statistical programmer will be performed.
- Critical (C): For datasets and outputs deemed critical, validation by an independent (i.e. the owner of the object and the validator of the object are different persons) statistical programmer or statistician will be performed. This may be a check against another validated source or a subset of another validated source/object (e.g. a figure produced in support of a summary table may be checked against that table). Validation does not have to be double programmed.
- Non-Critical (NC): For datasets and outputs deemed non-critical, validation in the form of manual review (e.g. object review or code review) by an independent (i.e. the owner of the object and the validator of the object are different persons) statistical programmer or statistician will be performed.

Once the validation level of objects has been set, a tracking sheet describing the activity of the team working on the project needs to be compiled. This document usually includes (both for datasets and outputs) for each object:

- validation level
- priority
- main programmer
- status (ongoing / ready for validation / final)
- date linked to the status
- program used for creating the object
- validation programmer
- validation status (fail / ongoing / validated)
- date linked to the validation status
- program used for validating the object
- comments

Both main and validation programmers keep this documents updated to let each other know the state of art of each object and to exchange information with the "comments" column, which mainly contains the findings arisen by the validators.

When the objects to be compared have validation level MC, once the double programming has been performed several procedures exist for comparing results produced, but most of them does not take in account all aspects they should.

After an overview of the common methods used, our way of proceeding is explained, highlighting the most useful aspects of it.

COMPARING TWO DATASETS

PROC COMPARE in SAS System® is the most common procedure for comparing two datasets.

Anyway, most of the time, some aspects in this procedure are difficult to check (e.g. missing observations, missing variables) or the result does not help sufficiently to understand where the difference is. If the datasets do not have the same number of rows, the comparison of values in the same variable is out of phase and to understand where

the differences are could be really challenging. Moreover, the output is huge and the risk of getting lost in it is high. Summarizing, most of the users find its output not user-friendly, especially when there are differences in datasets.
Other times some differences are highlighted when not useful, let's think for example to the difference in the umpteenth decimal value due to rounding.

OUR SOLUTION: THE MACRO %DBCOMPARE

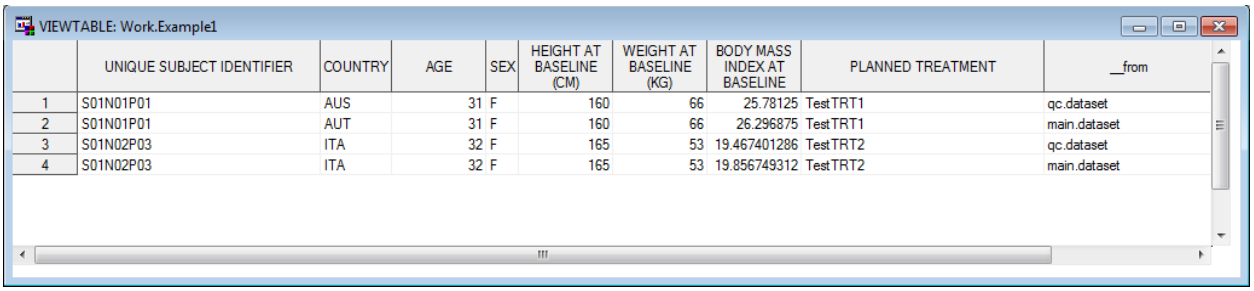
The macro %dbcompare we are using has many advantages that can be summarized by steps.
Firstly, with PROC CONTENTS, this macro allows the user to check if the datasets in input are comparable. A list of variables present in one but not in the other dataset are listed.

```
WARNING: the data sets are not comparable because they
         have different variables
VARIABLE=TRTEDT  LENGTH=8  FROM:main.dataset
```

Anyway, these differences (if any) do not stop the macro that goes through comparing the format and length of variables.

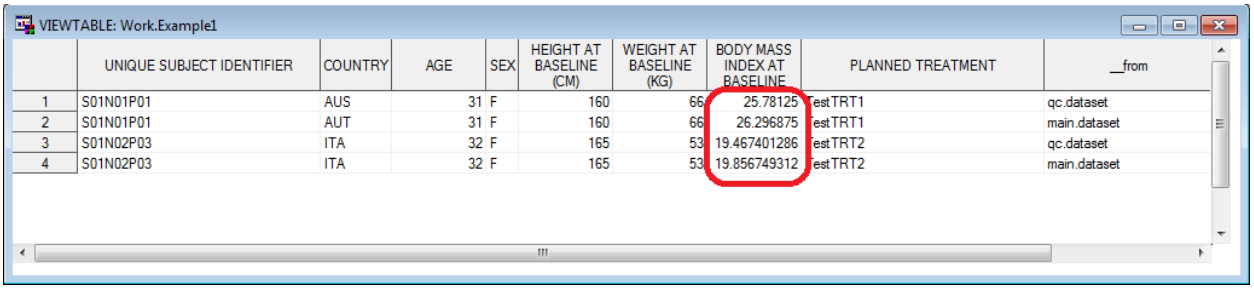
```
ERROR: the data sets are not comparable because they
       have different types:
VARIABLE=TRTSDT  TYPE=NUM  LENGTH=8  FROM:main.dataset
VARIABLE=TRTSDT  TYPE=CHAR  LENGTH=10  FROM:qc.dataset
```

Both information are clearly outputted in the log of SAS System® as shown above.
Once the differences in types are fixed we can go on with the last step of the macro.
This consists in showing in a dataset the observations without an equal counterpart in both datasets, specifying from what dataset they come from. This allows the user to simply order this output dataset to check the differences between observations, as shown in the below example.


VIEWTABLE: Work.Example1

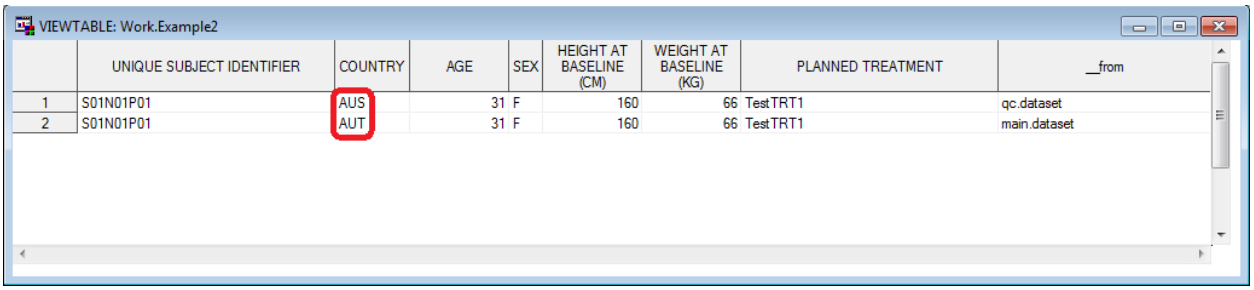
	UNIQUE SUBJECT IDENTIFIER	COUNTRY	AGE	SEX	HEIGHT AT BASELINE (CM)	WEIGHT AT BASELINE (KG)	BODY MASS INDEX AT BASELINE	PLANNED TREATMENT	__from
1	S01N01P01	AUS	31	F	160	66	25.78125	TestTRT1	qc.dataset
2	S01N01P01	AUT	31	F	160	66	26.296875	TestTRT1	main.dataset
3	S01N02P03	ITA	32	F	165	53	19.467401286	TestTRT2	qc.dataset
4	S01N02P03	ITA	32	F	165	53	19.856749312	TestTRT2	main.dataset

Moreover, this macro has two really important skills.
To avoid a lot of differences in the output dataset, the user can input the variables that he/she wants to exclude from comparison. For example, once we find a difference in the calculation of a score, it is useful to remove the differences generated by this and check for other differences one cannot notice because of too many errors in output.
The following example shows this situation:


VIEWTABLE: Work.Example1

	UNIQUE SUBJECT IDENTIFIER	COUNTRY	AGE	SEX	HEIGHT AT BASELINE (CM)	WEIGHT AT BASELINE (KG)	BODY MASS INDEX AT BASELINE	PLANNED TREATMENT	__from
1	S01N01P01	AUS	31	F	160	66	25.78125	TestTRT1	qc.dataset
2	S01N01P01	AUT	31	F	160	66	26.296875	TestTRT1	main.dataset
3	S01N02P03	ITA	32	F	165	53	19.467401286	TestTRT2	qc.dataset
4	S01N02P03	ITA	32	F	165	53	19.856749312	TestTRT2	main.dataset

We can easily notice from the output dataset a difference in the calculation of Body Mass Index score.
Once we take note of this we can run again the macro excluding that variable, obtaining as output:



Now that BMI differences have disappeared, we can notice another difference, this time on Country variable.

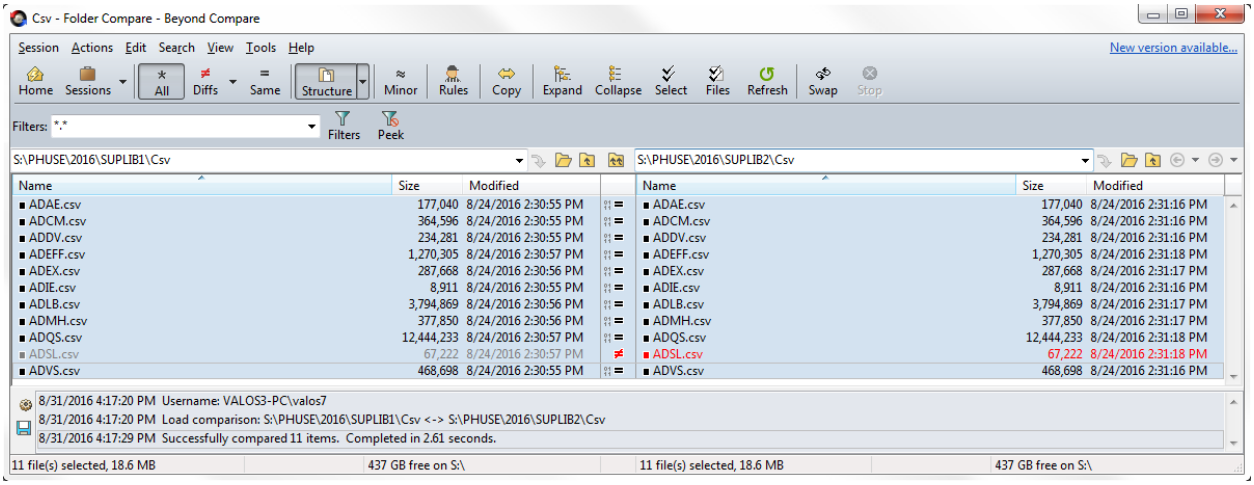
The other important skill is the possibility of comparing not only two datasets, but also two whole libraries. This is possible if the datasets to be compared inside the libraries are called with the same name. For each pair of datasets with the same name, the macro creates a different output dataset.

STILL PROBLEMS? THE LAST SOLUTION: BEYOND COMPARE®

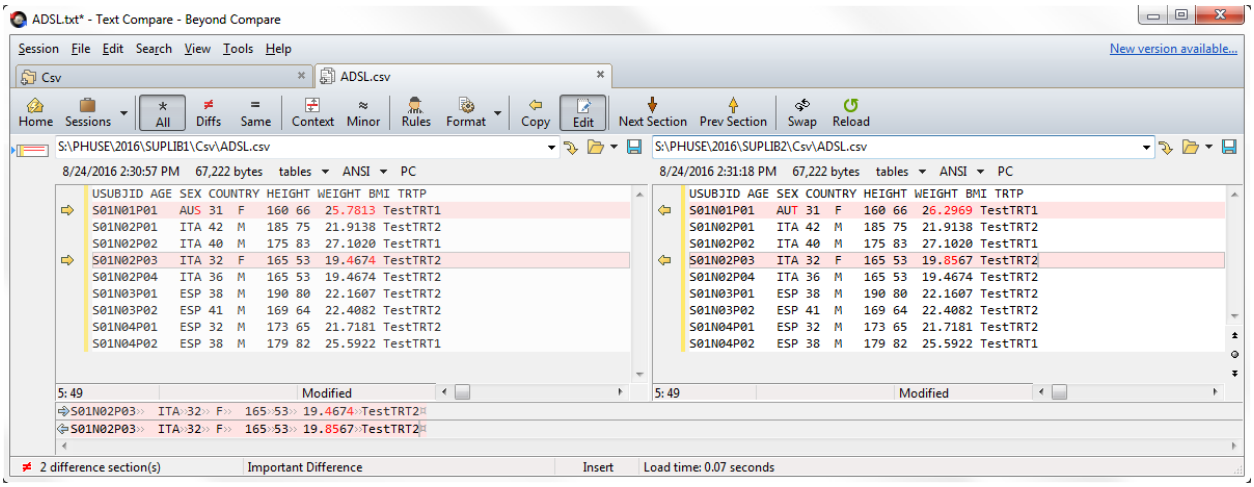
Sometimes it happens the %dbcompare macro outputs observation that seems to be equal and the user cannot find what the difference is. This can occur because of difference in an umpteenth decimal value that SAS System® rounding differently, based on the computer the user is using. If this is the case or if you simply want a good alternative, the definitive solution is to use Beyond Compare®.

Beyond Compare® allows the user to perform directly comparison between folders, text, and other types of file.

Once we exported the datasets to be compared (for example in a txt format) we can compare these with this program.



The rows with the symbol ≠ highlighted in red are the datasets with differences. Clicking on them a new tab will appear that allows the user to check for these differences.



In the above example the same two differences found with %dbcompare are discovered by Beyond Compare®, with the big advantage that this comparison is not affected by SAS System® rounding problem because it compares only what is displayed.

COMPARING TABLES, LISTINGS AND GRAPHS

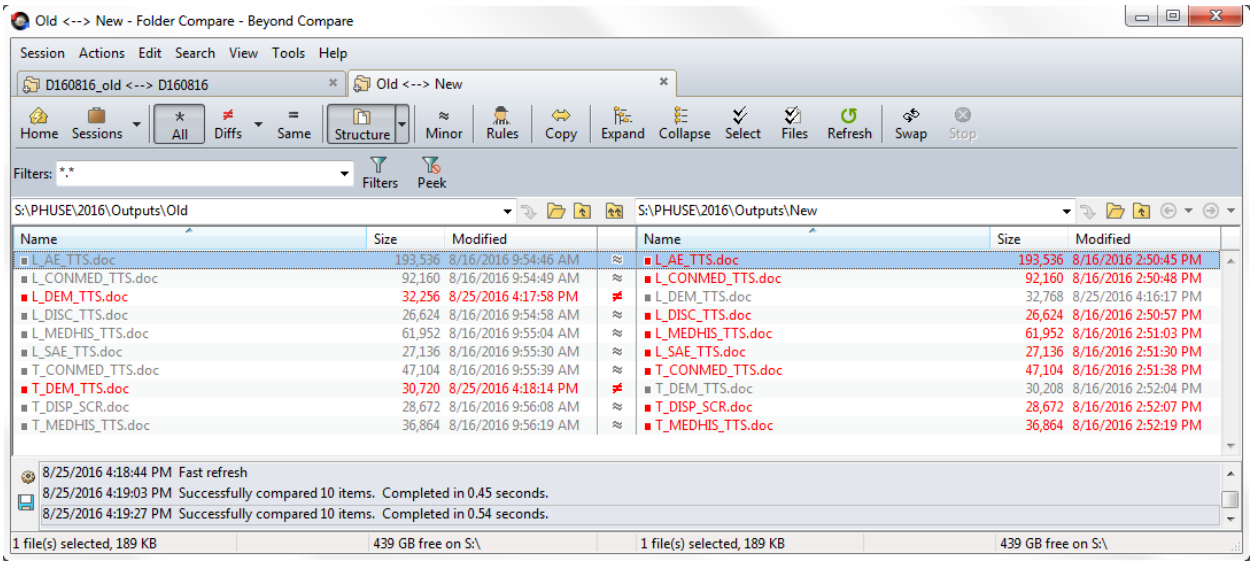
Depending on the format tables and listings are produced, different ways of comparing these outputs exist. Not to mention the countless errors one cannot find comparing tables and listings simply looking at them, we think the best way for perform the validation is to produce tables and listings in Microsoft Word® or Rich Text formats and check them with a program that permits to compare all the outputs automatically and mark only the outputs with different values.

Despite the large use of SAS System® PROC REPORT, we think this is not the best for producing outputs to be than compared. We created for this purpose a macro that allows the user to create tables and listings completely in text characters. As an example, the shell table below shows the baseline characteristics of subjects:

```
Study: S01
<Title>
Analysis set: <analysis set>

                                          Treatment A      Treatment B        Total
                                            N = XX           N = XX          N = XX
-------------------------------------------------------------------------------------

Age at screening (years)
  n                                           xx               xx              xx
  Mean                                        xx               xx              xx
  SD                                          xx               xx              xx
  Median                                      xx               xx              xx
  Q1, Q3                                   xx, zz           xx, zz          xx, zz
  Min, Max                                 xx, zz           xx, zz          xx, zz

Sex [n (%)]
  Male                                    xx (yy.y)        xx (yy.y)       xx (yy.y)
  Female                                  xx (yy.y)        xx (yy.y)       xx (yy.y)

Height (cm)
  n                                           xx               xx              xx
  Mean                                        xx               xx              xx
  SD                                          xx               xx              xx
  Median                                      xx               xx              xx
  Q1, Q3                                   xx, zz           xx, zz          xx, zz
  Min, Max                                 xx, zz           xx, zz          xx, zz

Weight (kg)
  n                                           xx               xx              xx
  Mean                                        xx               xx              xx
  SD                                          xx               xx              xx
  Median                                      xx               xx              xx
  Q1, Q3                                   xx, zz           xx, zz          xx, zz
  Min, Max                                 xx, zz           xx, zz          xx, zz

BMI (kg/m^2)
  n                                           xx               xx              xx
  Mean                                        xx               xx              xx
  SD                                          xx               xx              xx
  Median                                      xx               xx              xx
  Q1, Q3                                   xx, zz           xx, zz          xx, zz
  Min, Max                                 xx, zz           xx, zz          xx, zz

-------------------------------------------------------------------------------------

BMI: Body Mass Index  = Weight [kg] / (Height [m])^2

Table xxx - Produced by XXX on ddmmmyyyy hh:mm (CET), Data extraction date: ddmmmyyyy
Program: program_name.sas
Page x of x
```

USING BEYOND COMPARE® FOR TABLES AND LISTINGS

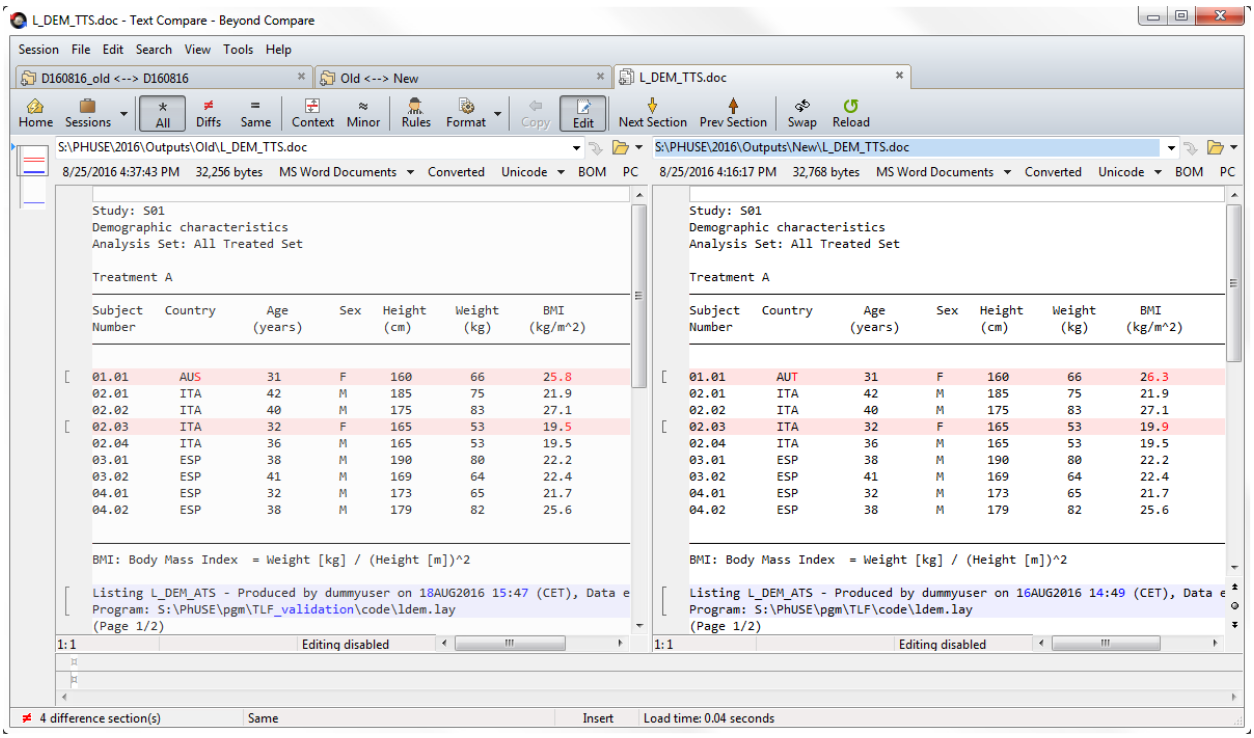
As shown for the dataset comparison, Beyond Compare® is really simple to use.



Once the user performs the comparison of outputs, three symbols are shown:

- = means that outputs are identical
- ≈ means that outputs are identical not considering the exceptions we put in the program
- ≠ means that outputs are different

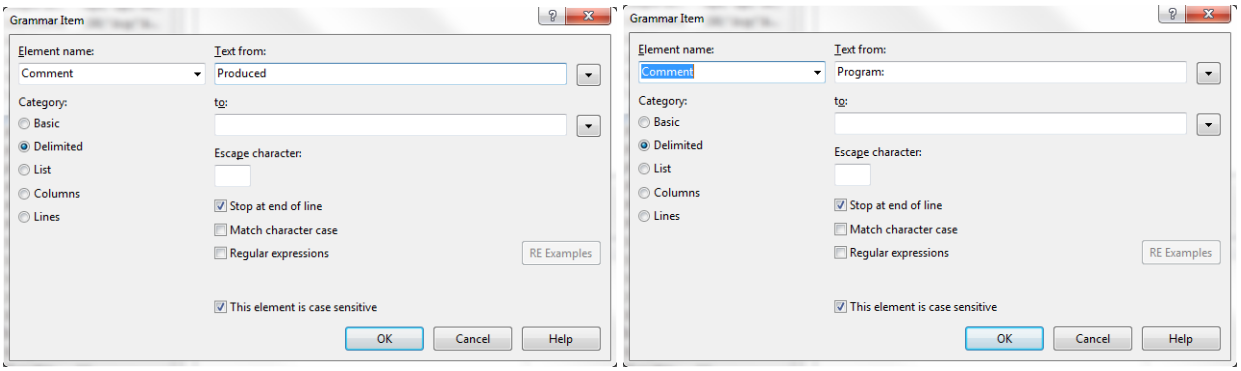
Following the example done in dataset comparison section, we notice both main listing and table of demographic characteristics are different to their validation version.
Clicking on the error symbol of the listing, the following screen will appear:



The rows with differences are highlighted in red.
At the top left of the screen there is a sort of summary indicating the differences in the rows:

- Red rows indicate the differences
- Blue rows indicate the exceptions

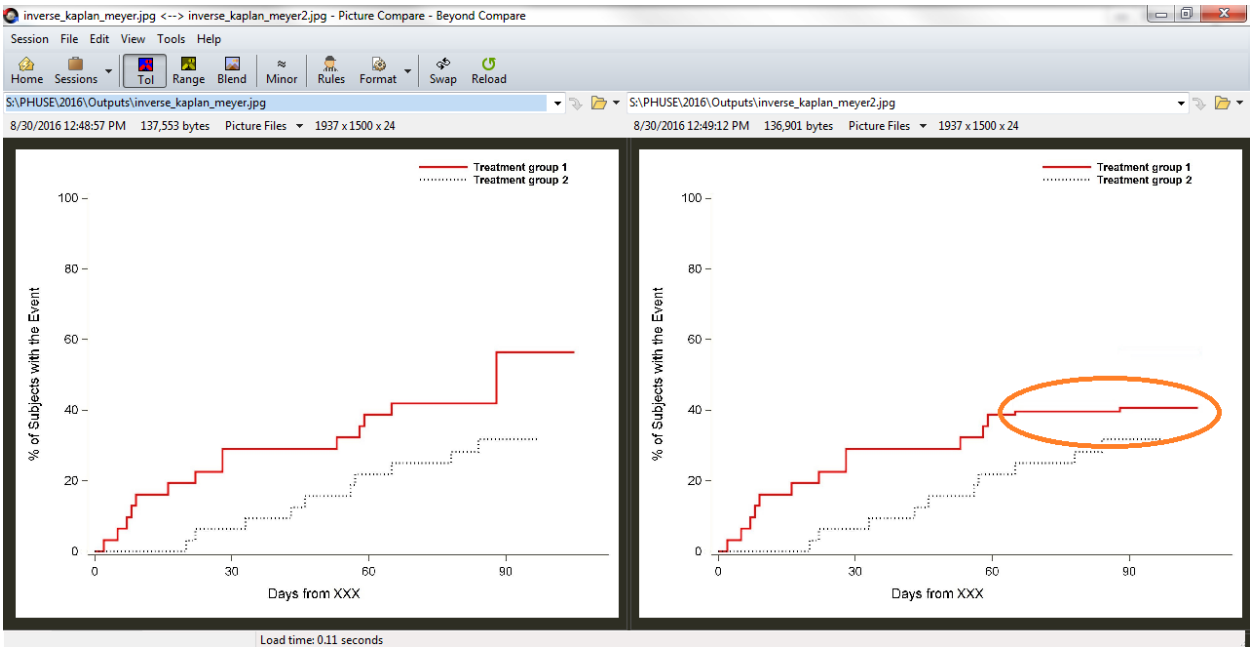
The exceptions are a useful facility for avoiding the program to output as errors all that difference that we don't want to consider. The above example includes as exceptions all the lines starting with the words "Produced" or "Program": this way all the differences in user, date and time of production and name of programs will not be considered.



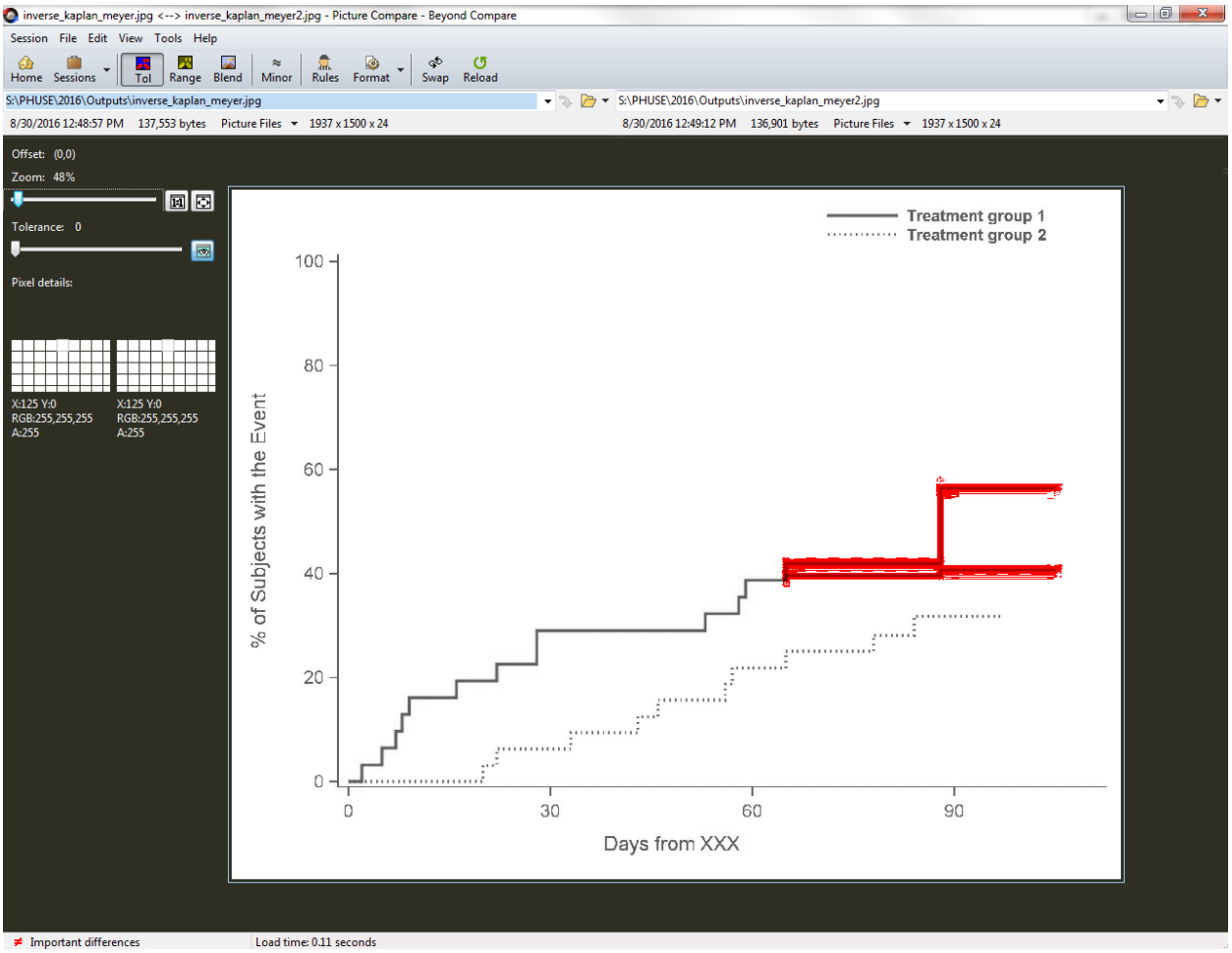
As shown in the above pictures, the user has a huge amount of options to choose for setting up the exception. These two are the settings applied in the example of the demographic characteristics table.

USING BEYOND COMPARE® FOR GRAPHS

Beyond Compare ® is a useful tool also for comparing graphs. In the example below, we can clearly see a difference in the red Kaplan-Meier curve.



Beyond Compare ® processes the graphs and outputs a new graph with the differences marked in red:



Even if usually the graphs are compared with tabular data and double programming is not really used for this purpose, this featuring is useful to perform comparisons between two different deliveries (for example for Data Monitoring Committee) to check what changed in the new delivery compared with the old one.

CONCLUSION

Programmers in charge of validating datasets and outputs must be aware that they not only have to focus on the outputs to be produced, but also on the way they have to compare them to the main outputs.

This way must be as much as possible free of errors. The best and less time-consuming method for us is the one described in this paper.

CONTACT INFORMATION

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

Author Name	Andrea Parodi
Company	Valos
Address	Via Antonio Gramsci 1/13
City / Postcode	16126
Work Phone:	+39 010 253 4160
Fax:	+39 010 253 4160
Email:	info@valos.it
Web:	www.valos.it