PharmaSUG 2024 - Paper SD-431

inspectoR: QC in R? No Problem!

Steve Wade, Sudhir Kedare, Chen Yang, Matt Travell and Jagan Mohan Achi, Jazz Pharmaceuticals, Inc.

ABSTRACT

More organizations are starting to embrace open-source technologies to perform tasks traditionally completed in SAS®. One such activity is to QC datasets, tables, and figures in the process of producing TLF's. Independent programming is done for many of those TLF's, comparing the results from both programmers. Jazz has developed the inspectoR package, an alternative to SAS/COMPARE® procedure, to allow QC performed in R to be compared back to datasets coming from a SAS system.

In this paper, we demonstrate how inspectoR will compare these datasets and produce a report showing the findings. The report produced by inspectoR is much like PROC COMPARE output but is produced in HTML in a more readable format.

inspectoR has proven to be a valuable tool in helping to transition QC tasks to R and maintain the level of quality expected from SAS systems.

INTRODUCTION

Jazz Pharmaceutical is a patient-focused growing Biopharma company undergoing rapid internal digital transformation and modernization of its data analytics capabilities while expanding into new molecule entities. The challenge to the newly formed Integrated Data Analytics and Statistical Programming group (IDASP) was how to begin to embrace open-source technology in a department that primarily uses SAS.

Within IDASP, the QC/validation process includes a technique in which a "production" programmer and a "QC" programmer independently use the same specifications to program an output. The QC programmer then compares their results to the production programmer's results. This entire process uses SAS with the QC programmer also using SAS/COMPARE. The output will typically be a SAS dataset such as SDTM, ADaM, or a dataset used to produce a report.

inspectoR was developed to provide an open-source tool that provides the same functionality as the SAS/COMPARE procedure. Using inspectoR, we can replace the QC side of the process with R. QC programmers can learn R by programming the dataset using R, then compare it back to the production side using inspectoR. Additionally, inspectoR improves the process by providing 1) a user-friendly, easy-to-read HTML report that indicates an overall pass/fail as well as individual test pass/fail statuses, and 2) a JSON file for each output with a pass/fail status used by the inspectoR status_report() function to create a status dashboard for all outputs. These improvements not only make the process faster and easier but also more efficient.

DETAILS

1. THE PROCESS

The approach was to use SAS for the "production side" and R for the "QC side" to achieve quality outputs in the independent programming process. Figure 1 shows the typical high-level workflow of the SAS independent programming process in the red boxes. The green boxes show the modifications when using inspectoR.

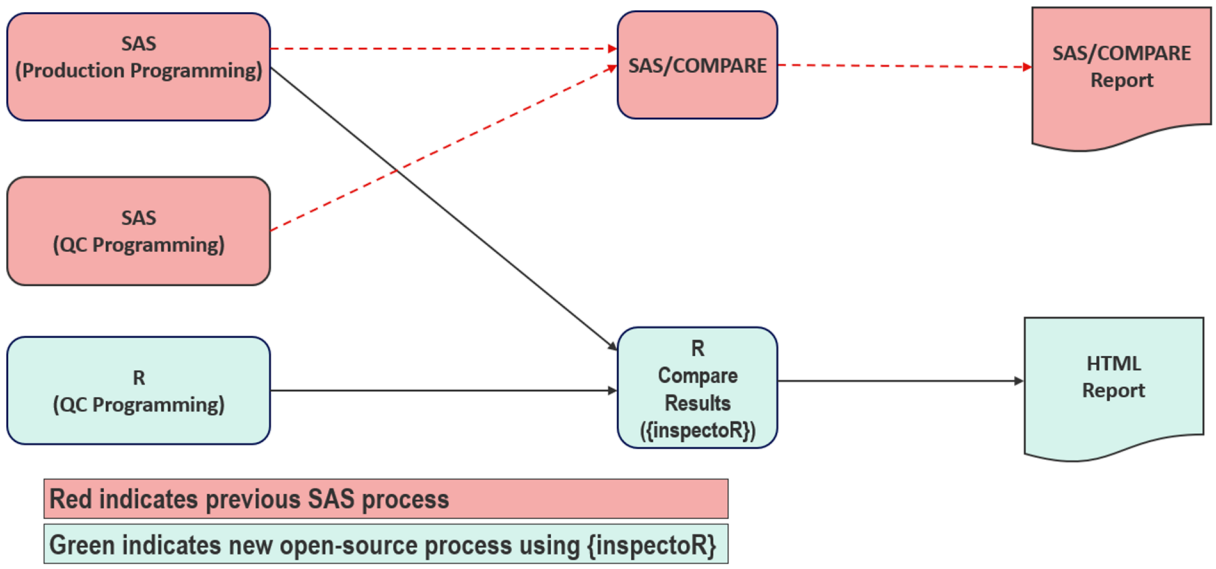


Figure 1. QC Process using open-source inspectoR package.

2. inspectoR FUNCTIONS

a) **compare_ds()**
b) **status_report()**

compare_ds()

Compare_ds is the main function of inspectoR that controls everything about the comparison. The output from this function is 1) an HTML pass/fail report, 2) a json file indicating pass/fail for the comparison.

compare_ds() function parameters:

Parameter	Default	Meaning	Example
base =		The first dataset to be used in the comparison	base = prod_ADSL
compare =		The second dataset to be used in the comparison	compare = QC_ADSL
base_label =	“Prod”	Descriptive label of first dataset. This is used in the HTML output.	base_label = “PROD”
compare_label =	“QC”	Descriptive label of second dataset. This is used in the HTML output.	Compare_label = “QC”
test_mode =	FALSE	If TRUE, outputs will be saved in a temporary location in the project	
generate_html_json =	TRUE	Whether to generate HTML and json output	

open_html =	TRUE	Whether to automatically open the generated HTML output	
output_loc =	NULL	Where you want the HTML and json files to be written	output_location = “study1\ADaM”
output_name =	NULL	The name you want given to the HTML and json files (without extension)	output_name = “ADSL”

Table 1. compare_ds() function parameters.

status_report()

The status report function will read all json files in a given directory and produce an overall pass/fail status report.

status_report() function parameters:

Parameter	Default	Meaning	Example
dir =	“.”	Directory where json files reside	dir = “study1\ADaM”
test_mode =	FALSE	If TRUE, report will be saved to temp location in the project	
output_dir =	“.”	Where you want to save the status report	output_directory = “study1\ADaM”
output_name =	“Status_Report”	The name you want to give to the status report	output_name = “ADaM status_04MAY2023”
open_html =	TRUE	If TRUE, report will be automatically opened.	

Table 2. status_report() function parameters

3. inspector USAGE

To accomplish the independent programming QC process, the production programmer will create a SAS program to produce a SAS dataset (prod_xyz). The QC programmer will create an R program to produce a dataset/dataframe using the same specifications (qc_xyz). Both the production and QC datasets should have the same structure, meaning the number, name, and order of variables should be the same as well as have the same basic type (numeric, character, date). The R program will then make a call to the inspectoR compare_ds() function to compare the two datasets:

```
compare_ds(base             = prod_xyz,
           compare           = qc_xyz,
           base_label        = “PROD”,
           compare_label     = “QC”,
           output_loc        = "/mylocation",
           output_name       = "xyz",
           open_html         = FALSE)
```

Currently, compare_ds will create a row number variable from one to the number of observations in each dataframe and compare by the row number variable. Allowing variables to be specified as keys is something for future development consideration.

The package compareDF is used to provide a cell-by-cell comparison of the two dataframes. This package produces a nice report highlighting differences in values between the two datasets.

4. inspectoR COMPARISON CHECKS

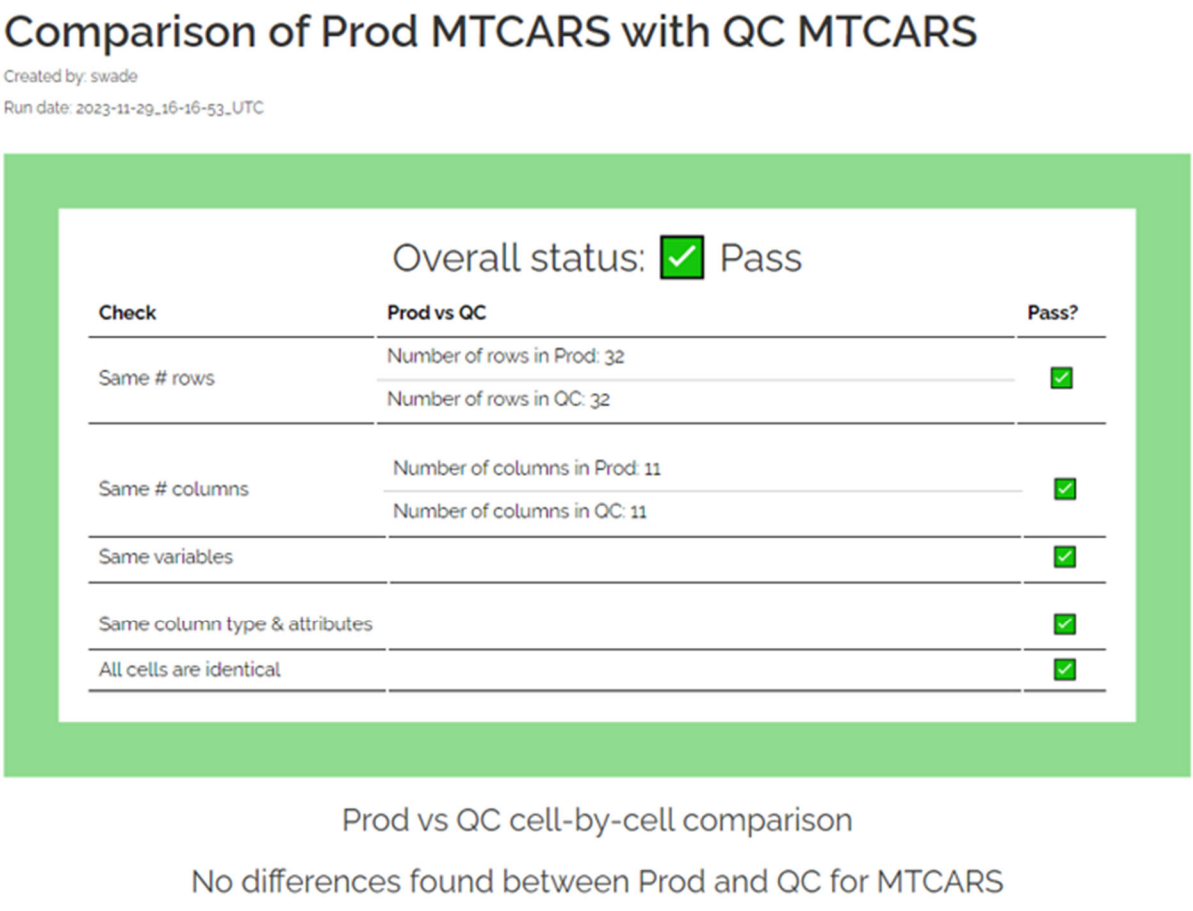
a. **Same # Rows –** Do the datasets contain the same number of observations?
b. **Same # Columns –** Do the datasets contain the same number of variables?
c. **Same Variables –** Do the datasets contain the SAME variables? If not, this will produce a report for 1) in "PROD" but not in "QC", 2) in "QC" but not in "PROD", or 3) both.
d. **Same Column Type & Attributes –** Do the variables have the same type (i.e. numeric/character/date) and/or attributes (i.e. labels)?
e. **Cell by Cell Comparison of Values –** Show differences between values, if any (compareDF package).

5. inspectoR EXAMPLE OUTPUT

The mtcars dataset from the R environment is used to show possible outcomes from inspectoR.

a. HTML Report:

Exact Match:



Display 1 HTML output example when datasets are an exact match.

Differences Found:

Comparison of PROD MTCARS with QC MTCARS

Created by: swade

Run date: 2023-12-20_22-05-23_UTC

Overall status: ✗ Fail

Check	PROD vs QC	Pass?
Same # rows	Number of rows in PROD: 32 Number of rows in QC: 32	✓
Same # columns	Number of columns in PROD: 9 Number of columns in QC: 10	✗
Same variables		✗

Variables in PROD but not in QC

Variable	Type	Label
wt	numeric	wt Variable

Variables in QC but not in PROD

Variable	Type	Label
am	numeric	am Variable
gear	numeric	gear Variable

Same column type & attributes		✗

Common variables with differing type or attributes

Variable	Dataset	Type	Attributes
carb	PROD	character	label : carb Variable
	QC	numeric	NULL

All cells are identical		✗

PROD vs QC cell-by-cell comparison

Showing the first 50 rows of differences

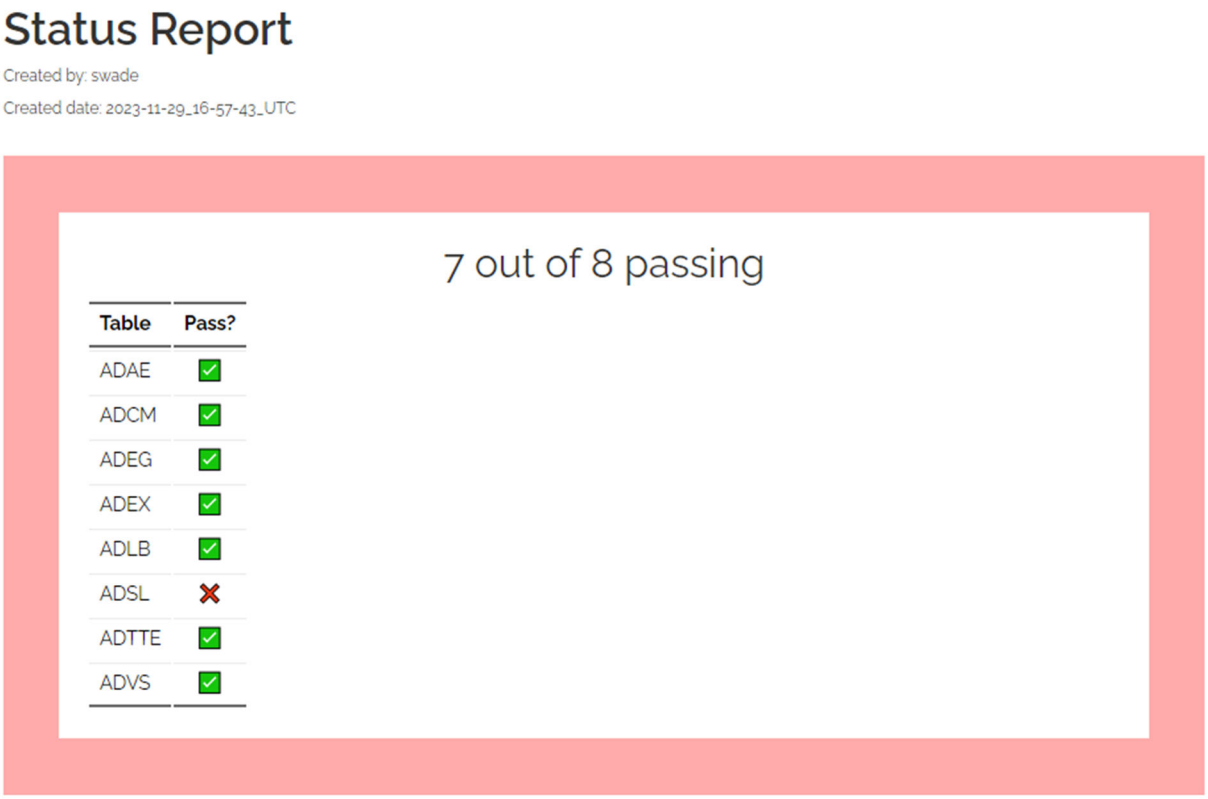
Row Number	Data From	mpg	cyl	disp	hp	drat	qsec	vs	carb
1	PROD	21	6	160	110	3.9	16.46	0	4
1	QC	30.9	6	160	110	3.9	16.46	0	4

Display 2 HTML output example when datasets have differences.

inspectoR, through the compareDF package, will report rows and columns from both datasets containing differences and color code the differences in each cell. The report produced by inspectoR is similar to PROC COMPARE output but is produced in HTML format and uses the format above for cell differences (Display 2).

b. Status Report:

In addition to the HTML output, a JSON file is also produced that contains information about the output and a pass/fail result. These JSON files can then be used to produce a dashboard in a “runall” type batch process to enable users to easily see what passed/failed for a given batch run. The status_report() function reads the json files and gives an overall pass/fail status of all datasets compared in a given directory similar to the output shown below.



Display 3 Status Report

CONCLUSION

inspectoR has proven to be a valuable tool in helping SAS programmers learn to program in R and maintain the level of quality expected from SAS systems. The HTML output is easy to read and user-friendly providing a fast, convenient way to see differences between datasets and the status report provides an overall status at a glance.

LESSONS LEARNED

R has a fairly steep learning curve coming from SAS. However, the new process gives SAS programmers a good way to learn R.

There are things to be aware of that can cause confusion:

- NA versus “” – R uses NA in character values can cause differences in the compare that appear as blanks. Setting NA values to “” is recommended on the R side to allow blank variables to compare equal.
- Some functions/merging in R may result in a blank at the end of a text value, which causes the values to be color-coded as differences when you cannot see the difference just be looking at it. There are techniques in R that can facilitate finding these differences such printing the values of a given variable to easily see one value has an extra space.

NEXT STEPS

- Continue to enhance the inspectoR package.
  - Allow the use of key variables.
  - Parameter to turn off attributes checks. For example, the R programmer may not have added variable labels, which will result in a lot of differences.
  - Limit number of HTML differences to show.

- Validate R to allow for more uses.

REFERENCES

compareDF Package: https://cran.r-project.org/web/packages/compareDF/index.html

CONTACT INFORMATION

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

Steve Wade
Jazz Pharmaceuticals, Inc.
swade@jazzpharma.com

Sudhir Kedare
Jazz Pharmaceuticals, Inc.
sudhir.kedare@jazzpharma.com

Chen Yang
Jazz Pharmaceuticals, Inc.
chen.yang@jazzpharma.com

Matt Travell
Jazz Pharmaceuticals, Inc.
matthew.travell@jazzpharma.com

Jagan Mohan Achi
Jazz Pharmaceuticals, Inc.
jachi@jazzpharma.com