ClinBAY
Statistical solutions to drug development

Executing R from SAS

ClinBAY Solution Series

Christos Stylianou, Ph.D

23-Jan-2020

ClinBAY Ltd
182 Agias Fylaxeos street
Office 101, Kofteros Business center
3083 Limassol, Cyprus

Table of Contents

Introduction

Different Statistical Packages have different advantages and capabilities. Whilst each customer has a package that is used on all their trials, it is sometimes the case that needs dictate a specific analysis to be performed that is not available in the aforementioned package.

Our client whose Statistical reporting is primarily performed using SAS, wanted to execute an analysis only available currently in R. The question was therefore how to streamline the reporting, i.e. to ensure consistency across the reports and make the transition between the packages seamlessly.

The go to approach is to check if the analysis only available in R could be replicated in SAS, sadly due to the complexity of the analysis this was not possible in this situation due to time and cost constraints. As it was clear that the reporting of the analysis should have been done in SAS to ensure consistent reports, another option that was explored was to do a stepwise production between the two packages. This approach was considered as both error prone and time-consuming as it would require 3 steps in our case, a run in SAS for preliminary analysis, a second step in R to produce the analysis and a final run in SAS to produce the reports.

It was therefore decided to explore the option to directly invoke R through SAS to ensure the production of the reports.

Methodology

The complexity of the problem was not only directly invoke R from SAS (as to ensure a single production execution), but also exchange of information between R and SAS. Despite SAS having a native method to invoke R through PROC IML, the SAS production environment in our case did not have PROC IML.

Before we proceed, it has to be noted, that both the native SAS functionality for R and the approach described here are running an R installed by the user. This means that if the production environment needs to be under a validated environment, then IQ/OQ/PQ validation will need to be performed in R separately to the IQ/OQ/PQ validation performed in SAS. However, this step is out of scope of this white paper.

Invoking R through SAS

SAS has a number of statements to invoke external applications. Out of the considered statements, using the “filename” statement with an unnamed pipe was selected. This approach enabled us invoke R through SAS and redirect the output to SAS without creating an intermediate data file. This enabled invoking the command prompt R and capturing the “console” for our Statistical analysis listing.

A note on this approach. When invoked in batch mode if the “pipe” seems to be idle for a number of minutes then SAS jumps to the next step, this could be an issue when running a computationally intensive method that could provide results after a number of minutes. The author has found no way to increase the idle time allowed in the pipe, however as the pipe is not really closed but idle, a request for SAS to sleep has the same results as long as the right sleep time is selected.

SAS code used for the invokation (full code in the Appendix)

```
filename proc_r pipe "&rfile CMD BATCH --vanilla --quiet  &rscript  &rlog" ;
```

Exchange of information

R has a number of packages available that can read SAS datasets (e.g. sas7bdat and haven) and write SAS datasets (e.g. haven and SASxport). However, there were a number of technical issues with using such a package for example having additional R package(s) to validate and packages falling into obscurity later due to migration to new SAS or R versions.

Therefore the solution was opted to use CSV files to exchange information between the two software as both SAS and R base have built in functions to write and read data, see proc export and proc import in SAS and read.CSV and write.CSV in R.

Finally, the log file generated by R was imported and printed in SAS.

Conclusion

This project demonstrated the potential to use different software packages working together efficiently in a single reporting environment.

Key Takeaways

1. Using a single reporting environment can ensure consistent reports even when using multiple software packages.
2. Requesting a single execution from a single package rather than multiple, created a dynamic environment which could efficiently handle co-operation between packages and avoiding multiple executions and transfer of information between the packages which could potentially have led to mistakes.

ClinBAY is a company that provides biometrics solutions to decision makers. Feel free to contact us (info@clinbay.com) if you are interested in our services or products.

Appendix: Sample SAS Code

General notes on the code:

- Datasets:
  - inputdata is the input file for R and is convered into a file named inputdata.csv
  - theRoutput.csv is a CSV file generated from R (for our reports) that is then imported in SAS as theRoutput dataset
- Macro variables:
  - rfile defines the R location and file to be invoked
  - execdir defines the location of the execution directory to be used for exchange of data between R and SAS
  - rscript defines the location and filename of the R script to be executed in R
  - rlog defines the location and filename of the log to be generated by R

Code:

```
%let execdir==%str(C:\Production);
%let rfile =%str(C:\R\R-3.4.3\bin\R.exe);
%let rscript=%str(C:\Production\myrscript.R);
%let rlog=%str(C:\Production\myrscriptlog.txt);
proc export data=inputdata outfile="&execdir/inputdata.csv" replace; run;
filename proc_r pipe "&rfile CMD BATCH --vanilla --quiet  &rscript  &rlog" ;
proc import datafile="&execdir/theRoutput.csv" out= theRoutput replace; run;
```