PhUSE 2016

Paper RW02

Real world evidence: a comparison between countries – Common data models and SAS macro catalogs

Edith M. Heintjes, Jetty A. Overbeek,
PHARMO Institute for Drug Outcomes Research, Utrecht, Netherlands

ABSTRACT

Despite European approval procedures and guidelines, country specific differences in local guidelines, reimbursements, organisation of care and availability of products may result in differences in treatment patterns across European countries. These patterns may inform marketing decisions, feasibility of effectiveness studies for real world evidence (RWE), or form the basis for such studies. To ensure comparability of results across countries, standardisation of analysis is necessary.

The PHARMO Institute conducted a study in five European countries regarding drugs used in type 2 diabetes. Database holders prepared input files according to a common data model (CDM). The purpose of this CDM was to standardise the format and content of the data, so a validated SAS® analysis tool could be applied to it. As local governance does not always allow exportation of patient level data, the tool was shared with the database holders and aggregated results were then shared for reporting purposes. To protect intellectual property we used a SAS stored compiled macro catalog. In the current paper we explain the concept of executing a multi-country study with SAS macro catalogs and discuss some of the options and possible pitfalls.

INTRODUCTION

RWE is playing an increasingly important role in pharmaco-epidemiology, in drug utilisation, effectiveness and post authorisation safety studies. In Europe, these studies often require a multi-country perspective. One problem encountered in the execution of such studies, is that despite development of similar protocols for each of the databases, the execution can in practice be very different. It then becomes difficult to interpret differences in outcomes of the studies across countries, because it is not clear whether differences represent real differences between the countries, or artefacts due to differences in execution. This is one of the reasons why CDMs are developed in European networks, to allow access to and analysis of data-sources from a central point. However, national legislation and governance rules of databases may prevent databases to allow access to or exportation of other than aggregated data to an external party. We describe how CDMs in combination with exportable SAS macro catalogs can be used to circumvent these problems, and discuss the possible pitfalls of this approach. We use our experience with a multi-country study as study coordinator at the PHARMO Institute of Drug Outcomes Research as an example to describe the process.

CONCEPT OF THE MULTI-COUNTRY STUDY EXECUTION

We performed a multi-country database study, using five years of retrospective data from five European databases. The aim was to determine treatment patterns of type 2 diabetes across countries and to determine which factors were associated with treatment choice. For all type 2 diabetes patients database custodians prepared input files for the study according to a CDM instruction. The input files remained at the site of the database custodian at all times. Summary statistics of the input files were shared with the study coordinator to check for problems with the prepared input files. Data processing and analyses were performed at each site by invoking macros from shared stored compiled SAS macro catalogs containing the SAS code prepared by the study coordinators. The resulting output tables with aggregated data were sent to the study coordinator for creation of the study reports. This approach, where only aggregated results were shared, avoided any problems with governance rules of each database that might have prohibited the sharing of patient level data. It also ensured that all data processing and analyses were done in exactly the same way at all sites. Hiding the code in the SAS macro catalog ensured that the intellectual property of the analysis tool was protected, despite sharing. We will go into some aspects of performing such a study using SAS macro catalogs.

SAS RELEASE AND OPERATING SYSTEM COMPATIBILITY REQUIREMENTS

Sharing SAS macro catalogs does require compatibility between the operating system and SAS release version in which the catalog is created and those in which the catalog is to be executed. Moving a SAS macro catalog from one release of SAS to another or from one operating system to another is not supported by SAS.

Attempting to use a SAS stored compiled macro catalog on a different operating system may result in the following error.

```
ERROR: File libref.SASMACR.CATALOG was created for a different operating system.
NOTE: The SAS System was unable to open the macro library referenced by the
      SASMSTORE = libref .
WARNING: Apparent invocation of macro  not resolved.
```

```
ERROR 180-322: Statement is not valid or it is used out of proper order.
```

We refer to SAS Usage Note 46846 for a list of possible operating systems [1].

Attempting to use the SAS macro catalog on a different release may also result in an error. In our experience it was possible to share a catalog created in SAS version 9.2 with database custodians using version 9.2 or higher up to 9.4, when all parties are using the same operating systems.
It is therefore advised to check compatibility with a simple test catalog, to check if systems are compatible at the start of the planned study. When performing this check, do not be satisfied with the finding that the SAS macro catalog will execute. You should also check if the resulting datasets can be read by the study coordinator, to avoid unexpected problems later on. For example, we found that a catalog created in SAS version 9.2 could be executed by SAS version 9.4, but that the tables that were sent back could not be read by version 9.2. This was due to a new system option EXTENDOBSCOUNTER or EOC that was introduced in version 9.3, and caused the following error when trying to open transferred SAS files created in version 9.4.

```
ERROR: File libref.member.DATA not compatible with this SAS version.
```

This system option is set default to yes in version 9.4. The problem can be solved by including the following system option in your execution macro in the SAS macro catalog (see example in section “SETTING UP COMMON PROGRAMMING CODE IN A CATALOG”).

```
EXTENDOBSCOUNTER = no;
```

COMMON DATA MODEL

In order to be able to apply the same programming to all databases, a CDM is necessary. A code book with predefined variable names, formats and definition of contents should be prepared by the study coordinator. Each database custodian will be responsible for data extraction and creation of input files according to the instructions in the code book. Each database custodian can use their knowledge of their database to determine the best method of creating the predefined variables, using the relevant coding systems used in their electronic health records (ICD-9, ICD-10, ICPC or any other relevant coding system) or other available information (text mining), as deemed appropriate by the database custodian, and as agreed with the study coordinator. In our example study these included a patient file with patient ID number, demographics and enrolment start and end date, a medication file with all prescriptions of antidiabetic medications, and an assessment file with all clinical assessments and diagnoses (e.g. HbA1c, BMI, blood pressure, chronic kidney disease). Patient ID was the key variable used for linking all input files. The medication and assessment files included multiple time stamped records per patient ID.

DATA DUE DILLIGENCE

It is imperative that you check with all database partners the contents of all possible input parameters to make sure that all possible entries can be handled correctly by your programs.

DATA SUMMARIES

We advise you to compare summary statistics between all databases prior to execution of the study, to make sure that all databases use the same units (e.g. pounds or kilograms, HbA1c % or mmol/l), to make sure there are no values outside the predefined accepted range for specific variables (e.g. outliers that do not make clinical sense) and to make sure all categories present in the databases are covered by the programming (e.g. some countries may use codes for medicines that diverge from the World Health Organization’s Anatomical Therapeutic Chemical (ATC) classification system).

BUILD IN CHECKS IN YOUR CATALOG

We advise you to build in data checks on the input datasets in your catalog, which will prompt error messages that will point out the problem, and will stop the execution until the problem is fixed. You may want to check the logical relations between variables that cannot be caught by the summary statistics.

CREATING A SAS MACRO CATALOG

A SAS macro catalog may contain a large number of compiled stored macros, facilitating multiple users at one site to access the compiled macro, thereby saving time and ensuring everyone uses the same version of a macro. SAS catalogs will also allow you to transfer a set of compiled macros as a single file with another site, such as a foreign database custodian. This catalog can contain all the programming code for all data processing and analyses of a study, as well as an execution macro to make sure the macros contained in the catalog are invoked in the right order (see section “Setting up common PROGRAMMING CODE in a catalog”.

DEFINING THE LOCATION OF THE SAS MACRO CATALOG

In order to create your own SAS macro catalog, you first need to define the location where this catalog is to be stored.

```
options mstored sasmstore=TPcat;
libname TPcat 'Directory:\folder\subfolder(s)\Catalog_TP' ;
```

Users at other sites will need to execute the same statements, but they will need to adapt the path in the libname statement to describe the location where the shared catalog is stored.

STORING A MACRO IN A CATALOG

The following example shows how to store a macro in a SAS macro catalog. In general it may be very useful for users of the catalog to be able to view the source code. It will enable them to check whether the analyses are correctly performed. In order to allow others to view the source code, each of the compiled macros stored in the catalog should be stored as in the example below:

```
%macro macro_name (parameters) / store source;
    macro program code
%mend macro_name;
```

The default name of the SAS macro catalog will be *sasmacr*. In order to view the contents of the catalog the following statements can be used.

```
proc catalog catalog = tpcat.sasmacr;
    contents;
quit;
```

This will result in a list of all the macros stored in the catalog. The following statement can be used to actually view the source code of a specific macro in the catalog.

```
%copy macro_name / source;
```

We suggest reading Stojanovic & Hollis (2005) and Walega (1996) if you wish to save this information into a file. For the purpose of multi-country studies we strongly suggest only the study coordinator is allowed to make changes to the program.

Source code can also be hidden in a catalog, creating a 'black box' that will execute the desired analyses. This may be useful if you want to protect intellectual property of a developed analysis tool that you do not wish to share with everyone for use in other studies. In order to hide the programming code for external users, apply the following syntax when storing each of your macros:

```
%macro macro_name / store;
    macro program code
%mend macro_name;
```

If you do not include the source code in your catalog, the source code will also not show in your log file. Only ERRORs and WARNINGs will be visible in the log-file.

USEFUL SYSTEM OPTIONS

The SAS macro catalog will allow you to execute a large number of data steps and analyses, potentially creating a large log file. This large log file may actually hamper execution of the analyses. In order to prevent large log files from being created, the following system options statement may be used.

```
options nonotes nomprint nomlogic nosymbolgen nosource nosource2;
```

This options statement can be incorporated as a first statement in your invocation macro in the catalog. In order of appearance these options will help to suppress writing of the following information to the log file:

- the notes associated with each data processing step, including the source code;
- display of the SAS statements that are generated by macro execution;
- tracing of macro execution (the beginning and end of macro execution, values of macro parameters at invocation, execution of each macro program statement, whether each %IF condition is true or false);
- the results of resolving macro variable references;
- source lines, except those that contain errors;
- secondary source statements from files that are included with an %INCLUDE statement.

The use of these options when the source code was included in the SAS stored compiled macro catalog will allow you to keep the code hidden if execution goes smoothly, but will also allow you to direct users to access the code, change the options and thus facilitate trouble shooting. Changing the actual source code should always be done by the study coordinator, to make sure all parties use the same version of the programs.

SETTING UP COMMON PROGRAMMING CODE IN A CATALOG

The CDM ensures all database custodians will be able to use common SAS programming codes and thus analyses will be identical between all databases. We recommend setting up the programming code in a modular way, using separate macros for each well defined step in data processing and analyses. If you aim to make your data available only for one specific study, you may also want to consider using a time sensitive key for access to the catalog, to prevent the use of it outside the period of study execution. To make sure all macros are executed in the intended order, we advise you to organise all modular macros into a final execution macro, which also contains required options, the optional time sensitive access key, and possibly also input data checks that might require the program to be stopped until inconsistencies or errors have been solved.

The processing and analyses codes are organised in a series of macro codes, to be invoked in specific order in a separate execution macro, along the following lines. All macro components, plus the execution macro catalog_tp that will ultimately invoke the macros are stored in the SAS macro catalog, in this example with hidden source code.

```
/* access key */
%macro access(key=) / store ;
     %let access=N;

     data _null_;
     date=today();
     call symput('date', date);
     run;

     /* 2013 Q1 */
     %if %sysevalf('01jan2013'd <= &date and &date <='31mar2013'd) %then %do;
          %if &key=t2UStiU0 %then %do;
             %let access=Y;
     %end;
     %end;
     /* 2013 Q2 */
     %if %sysevalf('01apr2013'd <=&date and &date <='30jun2013'd) %then %do;
     %if &key=qleBr8ew %then %do;
        %let access=Y;
     %end;
     %end;

     %if &access=N %then %do;
         %put ERROR: Access is denied, wrong key ;
         %abort;
     %end;

%mend access;

/* data processing macros*/
%macro step1 (parameters) / store ;
    Macro code
%mend step1;
%macro step2 (parameters) / store;
    Macro code
%mend step2;

/* data analyses macros*/
%macro step3 (parameters) / store ;
    Macro code
%mend step3;
%macro step4 (parameters) / store;
    Macro code
%mend step4;

/* execution macro */
%macro exec_tp(  RxDs = ,
                 PtDs = ,
                 endStudyPeriod = ,
                 startStudyPeriod = ,
                 inputdir = ,
                 outputdir =
```

```
                accesskey = ) / store;

    /* check if the access key corresponds with the provided list */
    %access(key=&accesskey);

    /* make sure files can be read by lower SAS release versions */
    Options EXTENDOBSCOUNTER = no;

    /* suppress log file notifications */
    options nonotes nomprint nomlogic nosymbolgen nosource nosource2;

    /* invoke processing and analyses macros in required order */
    %step1 (parameters);
    %step2 (parameters);
    %step3 (parameters);
    %step4 (parameters);

    /* reinstate log file notifications */
    options notes mprint mlogic symbolgen source source2;

%mend exec_tp;
```

EXECUTING THE PROGRAMS

In order to execute the macro to analyse the data, each user needs to fill in the appropriate pathnames and other macro variables that may differ per database, e.g. start and end of study period. The code to invoke the macro will be provided to the database custodian, along with instructions how to fill in the required macro parameters. Invoking the execution macro (example below) will then result in data processing and analysis, creating the output files to be shared and incorporated in a report.

```
%exec_TP          (RxDs = rxfile                          /* Prescription dataset name */
                  ,PtDs = patientfile                     /* Patient dataset name */
                  ,endStudyPeriod =   "31dec2011"d        /* End study period */
                  ,startStudyPeriod = "01jan2007"d        /* Start study period */
                  ,inputdir = Directory:\folder\input     /* Directory of input data */
                  ,outputdir = Directory:\folder\output   /* Directory of output data */
                  ,accesskey = qleBr8ew) ;
```

NOTES ON AUTOMATED DATA PROCESSING

Some analyses will depend on the actual data available. For example aggregation of treatments may depend on the frequencies of actual treatments in each of the countries, which may also change over time in the study period. Frequencies of occurrences of specific conditions can be used to set up decision rules for analyses and selection of variables. Therefore many analyses will make use of automated selection of variables, or automated grouping based on programmed decision rules. Automation of the data analyses will require making specific data-driven lists of variables (recommended literature Carpenter (2005)). This will require rigorous validation of the algorithms, to reduce the risk that unexpected differences in the data between databases will result in errors in your log-file, or - even worse - undetected errors in your analyses. We point out some specific aspects that are especially relevant with multi-country studies in this section.

CAPITALISATION

Capital use in variable names will usually not affect analyses. However, if you use automated variable lists for processing your data you may encounter problems if you do not account for capital use. Here is an example.
You can use ‘proc contents’ and ‘proc sql’ to automatically create a list of variables present in a dataset, and then use the %scan function to sequentially analyse those variables. Consider the following below using the following dataset *patient*.

PatID	Gender	Age	BMI	HbA1c
1	F	45	21	57
2	M	56	27	75
3	…			

In order to determine average age, BMI and HbA1c when not all variables may be present in all databases, it is necessary to only analyse those variables that actually exist. Automating this can be done as follows.

```
proc contents data = patient out = outfile (keep = varnum name);
run;
```

```
proc sort data = outfile;
by varnum;
run;

proc sql;
select name into: varlist separated by " " from outfile where name not in
("PatID","Gender");
select count(name) into: nvar from outfile where name not in ("PatID","Gender");
quit;
```

The macro variable varlist will now contain the following information as a character string:

```
Age BMI HbA1c;
```

The macro variable nvar will now contain the number of variables included in that string, i.e. 3.
Now select each variable in turn for analysis in a do loop:

```
%do i = 1 %to &nvar;
    %let var = %scan(&varlist,&i," ");
    proc means data = patient;
    var &var;
    output out = out&var mean = mean;
    run;
%end;
```

In this example something as simple as inconsistencies in the use of capitals in the variable names of the input files will result in errors, because the selection of these variables for creation of a macro variable containing a list of variables is case sensitive. The inclusion of character variables will obviously cause errors when processed as numeric variables. This can be circumvented either by using the upcase(*variable*) or lowcase(*variable*) function in the where statements creating the variable lists, or by making sure that capitalisation is made uniform at the start of processing the input files.

GRAPHICS

If you want to create uniform graphics for all participating database partners, we advise you to automatically store the graphics as an output file using the SAS output delivery system, rather than exporting or saving a file manually from the output window. The latter may cause differences in appearance of the graphics between databases, which may be bothersome if you want to publish results.

CONCLUSION

Although governance rules and legislation may prevent direct access to electronic health records by pharmaceutical companies and also prevent exportation of those health records out of the country, the use of common data models in combination with SAS macro catalogs offers an option to make sure analyses in multiple countries are performed in the exact same way. An added bonus of this option is that one can make use of the local expertise with the database and health care system to construct the input files that best suit the purpose of the study and to help interpret the results.

Providing access to the source code of a SAS macro catalog will allow the user check the actual processing of data and help trouble shooting in case of errors. The study coordinator can then change the programming for all parties if necessary. Hiding the source code to protect intellectual property is possible with SAS catalogs. However, development of fully automated execution macros using SAS stored compiled macro catalogs with out source code does require anticipation of all possible differences in the exact contents of the input datasets between countries and developing highly automated built-in data driven decision rules to deal with those differences correctly in the analyses.

Sharing SAS macro catalogs to analyse data across countries is a valuable tool to ensure identical analysis methods, while making use of the expertise of local investigators regarding the use of their database.

REFERENCES

1. Usage note 46846: Compatibility of stored compiled macro catalogs across releases of sas and across operating systems. URL: http://support.sas.com/kb/46/846.html

ACKNOWLEDGMENTS (HEADER 1)

We thank Eline Houben of the PHARMO Institute for Drug Outcomes Research for her input and critical review.

RECOMMENDED READING

Carpenter, Arthur L. (2005) “Storing and Using a List of Values in a Macro Variable” *Proceedings of the 30 Annual SAS Users Group International Conference*, paper cc02.

Stojanovic M., Hollis, D. “Ways to Store Macro Source Codes and How to Retrieve Them,” *Proceedings of the Southeast SAS Users Group* (2005), Paper AD07_05.

Walega Michael A.,(1996) "Creating, Maintaining, and Accessing SAS Macro Libraries," *Proceedings of the Ninth Annual Northeast SAS Users Group Conference*, 78-85.

CONTACT INFORMATION

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

Edith Heintjes / Jetty Overbeek
PHARMO Institute for Drug Outcomes Research
Van Deventerlaan 30-40
3528 AE Utrecht,
The Netherlands
Work Phone: +31 30 7440 800
Email: edith.heintjes@pharmo.nl
Web: www.pharmo.nl

SAS® product or service names are registered trademarks or trademarks of SAS Institute Inc., Cary, NC, USA.