PharmaSUG 2018 - Paper BB-12

SAS® automation techniques – specification driven programming for Lab CTC grade derivation and more

Yurong Dai, Yuqin Helen Li, Eli Lilly and Company

ABSTRACT

SAS programmers often face challenges on SAS coding, for example, Lab CTC grade derivation per lab test, putting variable's attribute into data sets – this can be very time consuming if we do this kind of work manually on coding, since the process repeats many times by test or data sets. Here I introduce a way for SAS coding in that SAS programming techniques take care of these repeated work and achieve automation, as long as a consistent pattern can be identified in the coding logic.

INTRODUCTION

Before programming starts, people spend tremendous amount of time on detailed programming guidance or derivation logic. When the logic is in a consistent format, automation on SAS coding is possible and this approach becomes my 1st choice.

This paper introduce examples that programming guidance documents become our source data, and are transformed into SAS macro - which can be repeatedly called in data derivation through transformer macros. Here I named the transformer macro labctcmaker, which essentially does: 1. a filename statement builds an output file directory and the file name. 2. read in the guidance document through data null, then output file labctc.sas which contains %labctc. The codes in the examples use LINUX system SAS Enterprise Guide 7.12.

1. LAB CTC GRADE'S DERIVATION FOR EXAMPLE 1

First we have CTC grade's derivation guidance document. An example can be as follows in Table 1, and only a few tests are listed here.

PARAMCD	ATOXGR
WBCA10S	IF AVAL < ANRLO then DO; If 3<=AVAL and AVAL<ANRLO then ATOXGR='Mild'; ELSE if 2<=AVAL and AVAL<3 then ATOXGR='Moderate'; ELSE if 1<=AVAL and AVAL<2 then ATOXGR='Severe'; ELSE if AVAL<1 then ATOXGR='Life threatening'; END; ELSE IF AVAL>=ANRLO then DO; IF ANRLO <= AVAL and AVAL <= 100 then ATOXGR='Normal'; IF AVAL > 100 then ATOXGR='Severe'; END;
NEUTA13S	IF AVAL>=ANRLO then ATOXGR='Normal'; IF 1.5<=AVAL and AVAL<ANRLO then ATOXGR='Mild'; IF 1<=AVAL and AVAL<1.5 then ATOXGR='Moderate'; IF 0.5<=AVAL and AVAL<1 then ATOXGR='Severe';

	IF AVAL<0.5 then ATOXGR='Life threatening';
LYMA14S	IF AVAL<ANRLO then DO; IF 0.8<=AVAL and AVAL<ANRLO then ATOXGR='Mild'; ELSE IF 0.5<=AVAL and AVAL<0.8 then ATOXGR='Moderate'; ELSE IF 0.2<=AVAL and AVAL<0.5 then ATOXGR='Severe'; ELSE IF AVAL<0.2 then ATOXGR='Life threatening'; END; ELSE IF ANRLO<=AVAL and AVAL<=4 then ATOXGR='Normal'; Else if 4<AVAL and AVAL<=20 then ATOXGR='Moderate'; Else if AVAL>20 then ATOXGR='Severe'; END;

Table 1. Lab CTC grade definitions without abnormality directions

Copying the contents in the above table and paste to SAS code can work, but it is time consuming. Since the original document is neat, what I did for editing the file was to add the case of missing values into the derivation logic before I used it. Then I use the following transformer macro %labctcmaker- The code creates a macro named %labctc, and in a SAS file named labctc.sas.

```
/*=============================================================================
This macro generate a new macro named %labctc for deriving lab CTC grade
Note: after %labctc is validated, do not replace it, unless there are changes in the source Lab CTC document file
*FEXIST Function: Verifies the existence of an external file by its fileref;
*FDELETE Function: Deletes an external file or an empty directory;
&macrolib is the directory for outputting file labctc.sas, which contains macro %labctc
&indata: the data set that contains the derivation logic
&labdata: the lab intermediate data set that will be used to derive CTC grade
=============================================================================*/
%macro labctcmaker(indata,labdata);

filename labctc "&macrolib./labctc.sas"; *Associates a SAS fileref with an
external file;
data _null_;
      set &indata end=last;
      *if fexist("labctc") then rc=fdelete("labctc");*this line is required
                                              for PC sas, must delete it in EG;
     length PARAMCD_ $10;
      PARAMCD_ = "'"||compress(PARAMCD)||"'"; *adding quotation marks;
      FILE "&macrolib./labctc.sas"  DELIMITER=' '  LRECL=256 pad;
      if _n_=1 then PUT @1 "%nrstr(%macro labctc;)"
                     /@1 "data &labdata; set &labdata;"
                     /@1 "%nrstr(length ATOXGR $16;)";

                   PUT /@1 "%nrstr(if PARAMCD=)" PARAMCD_ "then do;";
                   put @5 "%nrstr(ATOXGR=)"  ATOXGR;
                   PUT @1 "end;";

     if last then do;
                  PUT @1 "RUN;";
                  PUT @1 "%nrstr(%mend labctc;)"; end;
```

```
run;
%mend labctcmaker;
```

Suppose lab4 is the intermediate data set that is for CTC grade's derivation, the following statement is required to call %labctcmaker.

```
%labctcmaker(ctc, lb4);
```

From the above code, the following macro can be generated.

```
%macro labctc;
data lb4; set lb4;
length ATOXGR $16;

if PARAMCD='WBCA10S' then do;

IF .<AVAL< ANRLO then DO;
   If 3<=AVAL<ANRLO then ATOXGR="Mild";
   ELSE if 2<=AVAL<3 then ATOXGR="Moderate";
   ELSE if 1<=AVAL<2 then ATOXGR="Severe";
   ELSE if .<AVAL<1 then ATOXGR="Life Threatening";
END;
   else IF .< ANRLO<=AVAL<= 100 then ATOXGR="Normal";
       else IF AVAL>100 then DO;
             ATOXGR="Severe";
       END;
end;

if PARAMCD='NEUTA13S' then do;
    IF AVAL>=ANRLO>.  then ATOXGR="Normal";
IF 1.5<=AVAL<ANRLO then ATOXGR="Mild";
IF 1<=AVAL<1.5 then ATOXGR="Moderate";
IF 0.5<=AVAL<1 then ATOXGR="Severe";
IF .< AVAL<0.5 then ATOXGR="Life Threatening";
end;

if PARAMCD='LYMA14S' then do;

IF .< AVAL<ANRLO then DO
  IF 0.8<=AVAL<ANRLO then ATOXGR="Mild";
    ELSE IF 0.5<=AVAL<0.8 then ATOXGR="Moderate";
      ELSE IF 0.2<=AVAL<0.5 then ATOXGR="Severe";
        ELSE IF .<AVAL<0.2 then ATOXGR="Life Threatening";
END;
  ELSE IF .<ANRLO<=AVAL<=4 then ATOXGR="Normal";
     Else if 4<AVAL<=20 then ATOXGR="Moderate";
       Else  IF AVAL>20 then ATOXGR="Severe";
end;

…… (here many lines are omitted)


RUN;
%mend labctc;
```

The following code can call macro %labctc for creating the CTC grade variable ATOXGR, and this can be used in every study for CTC grade's derivation.

```
%include "&macrolib./labctc.sas";

%labctc;
```

Specification document can be provided in another format as in the example below.

2. LAB CTC GRADE'S DERIVATION FOR EXAMPLE 2

CTC grade's derivation guidance document can be in other formats. As long as a document is given in a neat and consistent way, we may adjust %labctcmaker, to create new %labctc. For example, another document is given in the following table 2 – lab abnormality direction is provided as well, and only a few *tests are listed here.*

PARAMCD	*Direction*	*LEFT_LIMIT*	*RIGHT_LIMIT*	*ATOXGRN*
WBCA10S	*L*	*3<=*	*<ANRLO*	*1*
WBCA10S	L	2<=	<3	2
WBCA10S	L	1<=	<2	3
WBCA10S	L	.<	<1	4
WBCA10S		ANRLO <=	<=100	0
WBCA10S	H		>100	3
NEUTA13S			>=ANRLO	0
NEUTA13S	L	1.5<=	<ANRLO	1
NEUTA13S	L	1<=	<1.5	2
NEUTA13S	L	0.5<=	<1	3
NEUTA13S	L	.<	<0.5	4
LYMA14S	L	0.8<=	<ANRLO	1
LYMA14S	L	0.5<=	<0.8	2
LYMA14S	L	0.2<=	<0.5	3
LYMA14S	L	.<	<0.2	4
LYMA14S		ANRLO<=	<=4	0
LYMA14S	H	4<	<=20	2
LYMA14S	H		>20	3

Table 2. Lab CTC grade definitions with abnormality directions

AS showed below for the transformer macro labctcmaker, the code was adjusted according to the derivation logic's new pattern, and a variable for abnormality direction is added as well.

```
%macro labctcmaker(indata,labdata);

filename labctc ftp "'&macrolib./labctc.sas'"; *Associates a SAS fileref with
an external file;
data _null_;
      set &indata end=last;
      by paramcd direction atoxgrn;
      length direction_ $3 PARAMCD_ $10;
    direction_ = "'"||compress(direction)||"'"; *to convert value from
                                                 L,H to 'L', 'H' format;
    PARAMCD_ = "'"||compress(PARAMCD)||"'";
```

```
      *if fexist("labctc") then rc=fdelete("labctc");");*this line is
                                 required for PC sas, must delete it in EG;

      *Specifies the current output file for PUT statements;
    FILE "&macrolib./labctc.sas"  DELIMITER=' '  LRECL=256 pad;
    if _n_=1 then
       PUT @1 "%nrstr(%macro labctc;)"
          /@1 "%nrstr(data &labdata; set &labdata;)"
          /@1 "%nrstr(length direction $1;)";
         if first.paramcd then
          PUT @3 "%nrstr(if paramcd = )" paramcd_ "then do;";
              PUT @5 "if " LEFT_LIMIT " AVAL " RIGHT_LIMIT
                    "%nrstr(then do; ATOXGRN =)"  ATOXGRN "%nrstr(;
direction= )" direction_ "; end;";
       if last.paramcd then
          PUT @3 "end;";

      if last then do;
                  PUT @1 "RUN;";
                PUT @1 "%nrstr(%mend labctc;)";
      end;
run;
%mend labctcmaker;
```

Again, suppose lab4 is the intermediate data set that is for CTC grade's derivation, the following statement is required to call %labctcmaker.

```
%labctcmaker(ctc, lb4);
```

From the above for new transformer macro labctcmaker, the following macro can be generated.

```
%macro labctc;
data &labdata; set &labdata;
length direction $1;
  if paramcd = 'LYMA14S' then do;
    if ANRLO<=  AVAL <=4 then do; ATOXGRN =0 ; direction= '' ; end;
    if 4<  AVAL <=20 then do; ATOXGRN =2 ; direction= 'H' ; end;
    if    AVAL >20 then do; ATOXGRN =3 ; direction= 'H' ; end;
    if 0.8<=  AVAL <ANRLO then do; ATOXGRN =1 ; direction= 'L' ; end;
    if 0.5<=  AVAL <0.8 then do; ATOXGRN =2 ; direction= 'L' ; end;
    if 0.2<=  AVAL <0.5 then do; ATOXGRN =3 ; direction= 'L' ; end;
    if  .< AVAL <0.2 then do; ATOXGRN =4 ; direction= 'L' ; end;
  end;
  if paramcd = 'NEUTA13S' then do;
    if    AVAL >=ANRLO then do; ATOXGRN =0 ; direction= '' ; end;
    if 1.5<=  AVAL <ANRLO then do; ATOXGRN =1 ; direction= 'L' ; end;
    if 1<=  AVAL <1.5 then do; ATOXGRN =2 ; direction= 'L' ; end;
    if 0.5<=  AVAL <1 then do; ATOXGRN =3 ; direction= 'L' ; end;
    if  .< AVAL <0.5 then do; ATOXGRN =4 ; direction= 'L' ; end;
  end;
  if paramcd = 'WBCA10S' then do;
    if ANRLO <=  AVAL <=100 then do; ATOXGRN =0 ; direction= '' ; end;
    if    AVAL >100 then do; ATOXGRN =3 ; direction= 'H' ; end;
    if 3<=  AVAL <ANRLO then do; ATOXGRN =1 ; direction= 'L' ; end;
    if 2<=  AVAL <3 then do; ATOXGRN =2 ; direction= 'L' ; end;
    if 1<=  AVAL <2 then do; ATOXGRN =3 ; direction= 'L' ; end;
    if  .< AVAL <1 then do; ATOXGRN =4 ; direction= 'L' ; end;
  end;
```

```
…… (here many lines are omitted)

RUN;
%mend labctc;
```

The following code can call macro %labctc to generate the CTC grade variable ATOXGRN and abnormality direction variable.

```
%include "&macrolib./labctc.sas";

%labctc;
```

From the above examples, as long as the specification are in a consistent format, transformer macro %labctcmaker can be converted into SAS executable macro %labctc.

3. CONVERT THE SPECS FOR VARIABLE'S DEFINITION TO VARIABLE'S ATTRIBUTE, AND CONVERT VALUE LEVEL METADATA TO CDISC VARIABLE'S VALUES

Please refer my paper **PharmaSUG 2017 - Paper DA03 for the details. In this paper,** Metadata driven approach utilizing SAS programming techniques for SDTM and ADaM data mapping was introduced. Variable attributes, format, variable names and their order and sorting order were grabbed from the data specification, and were atomically converted to data elements through our macros. This technique increases code's reusability, efficiency and consistency between data specifications and output data, and reduces re-work after data specification's update, during code development for SDTM mapping and ADaM datasets derivation

CONCLUSION

Automation is worth of our efforts and can be achieved through our programmers' exploration. The success of automation can save us tremendous amount of time and improve efficiency and quality.

ACKNOWLEDGMENTS

Thank Lixiang Larry Liu for reviewing this paper

CONTACT INFORMATION

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

Yurong Dai
Eli Lilly and Company
myydai@yahoo.com

Yuqin Helen Li
Eli Lilly and Company
li_yuqin_helen@lilly.com