More Time on Thinking, Less Time on Coding - An Efficient Approach in SAS EG

Charles Ling, Incyte, DE

ABSTRACT

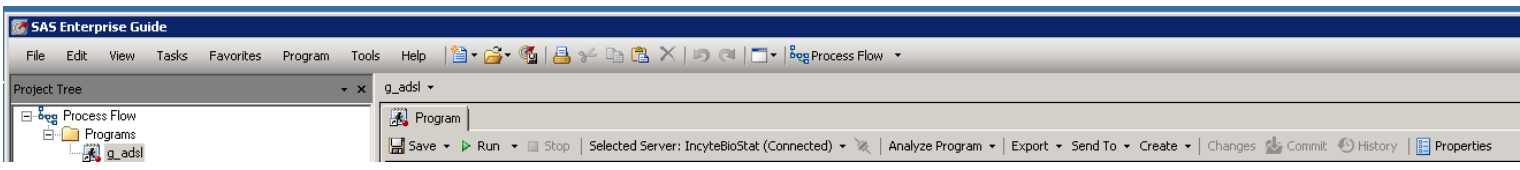
In general, SAS programmers follow and work with statisticians and the clinical team on a daily basis in a fast pace environment before a clinical study is complete. Therefore most of the programmers barely sit down and think about some important issues/questions: are the mock-ups make sense, do we interpret our primary/secondary endpoints correctly based on Protocol and SAP? Do the lay-out interpret the important questions that the study is expected. Do we produce too much unnecessary output or not enough on the key concepts? Do we fully understand the emails/meeting notes from the clinical team or do they fully understand our questions when we work in a fast-pace environment? When facing overwhelming tasks to finish, how can we reduce programmers' work by identifying some efficient way of coding on SAS EG (SAS EG is the only platform for discussion in this paper. Many key concepts presented in this paper are through demonstrations, rather than texts here).

Programmer's Tasks

We have three types of output in general: efficacy, safety and graphs. I am sure programmers do take some time at the very beginning of a study to go over protocols and SAP, and keep some notes on unclear concepts or even on TOC(table of contents). Some experienced programmers do save and reuse some old programs and make modifications to finish new assignments. But it takes time to look for similar code and can be time consuming. On SAS EG, I have discovered and tested, some major tasks can be accomplished with a form of semi-automation by creating macro buttons.

SAS EG

First, let us have a brief picture of the set-up on SAS EG.



After you log into your own project/study, there is a folder called “programs”, from which you may have several SAS programs to execute (TLFs/datasets). As a general practice, any programmer will begin a new code, starting from scratch, or copying some old code from another study to begin with, then add, modify, make changes until it achieves what is expected. Hours and hours are spent, some type fast, others do not, that is another issue. It is a pretty time-consuming task, on the other side, statisticians politely push for output, doctors can not wait for the results. However, there is an alternative way to achieve all by using a button-driven process.

Button-Driven Process

Step 1: select a TA that you work with most. Below is my example to share with you: (I created some of my own buttons, you may name any button you choose)

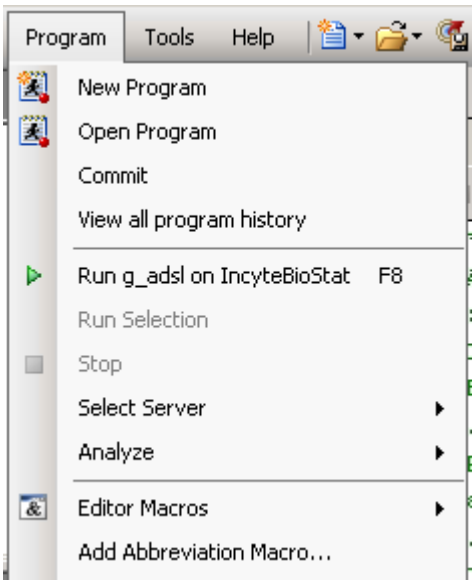
naming convention	XXXX_tab (demo_tab disp_tab adve_tab	Output #
aeout_tab	a sponsor-deined AE table, %aesum also used in this program	
ae_descending_tab	an AE table by descending counts	
ae_max_tab	an AE account of Max severity using %aesum	
aeterm	a list of common AE variables	
AGEGROUP	to calculate age groups	
age2	to calculate age (years)	
QC_listing4	AE listings	
demog_table	to create demog table	1.2.1
shift_tab	create shift talbels. 3.3.2	3.3.2
expoure_tab	to create exposure table, 2 types of tables	exposure and compliance
figure_lg_43	to create line graph	
array	use array	
attr	to compare Primary ADaM dataset with QC dataset	
blankshell	to create 5 columns with 7 dummy value	
comparevars	compare derived and QC variables	
comp_tab	to create compliance table	3.1.1
char2num	from sdtm.charVAR to date9. date var	
comparevars	compare derived and QC variables	
compare3vars	to compare over 2 variables only	

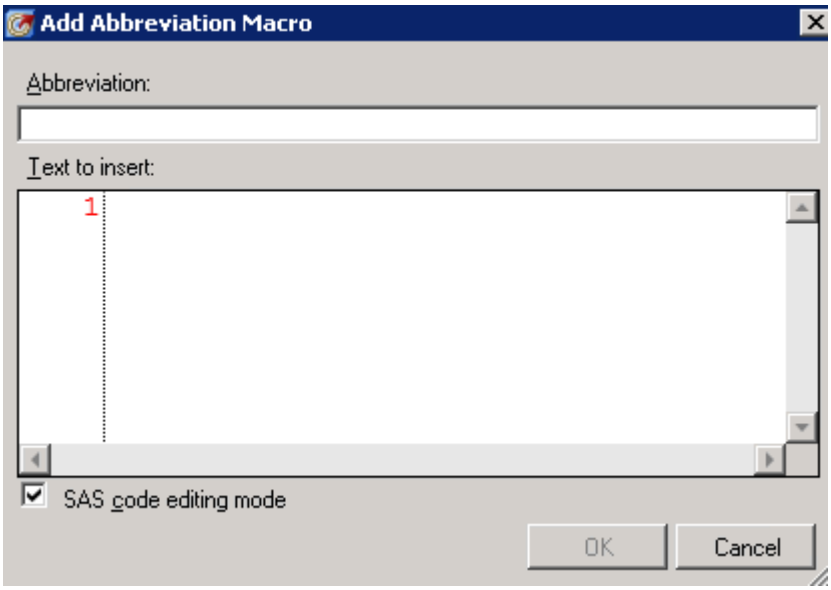
(Note: you design your own buttons)

You may create an excel sheet like the above one. In the first column, you list the button names; in the second column, you add some contents that a new button can achieve, and in the third column, you may add conventional output number, each company may have its own standard. For me, for some complicated concepts, I create buttons like “blankshell” to output a dummy table shell. You can design an excel sheet to fit your own needs. Now let’s see how the buttons are created.

Button Building

You may start from "Program"





1) You give a name like "AE_Overall" in the Abbreviation box, then add your well executed AE code next to "1". Click ok, a table is waiting for you to be executed.

2) Open your SAS program with header or not, I usually copy and past 10 programs according to my TOC. In the program, I simply type “AE_Overall”, you highlight the same words in the pop-up window, your code automatically appears. If your code does not need any modification, just hit run and feel free to grab a coffee (or take a 5-minute walk if you would like to).
3) Back to the excel sheet, you may add as many buttons as you want, plus lots of concept buttons, all related to your study and record them in the excel sheet.

One obvious question, can I modify these buttons? Yes.

Go to “Editor Macros/Macros

Select your button

Edit

Double click contents under “Keyboard macro contents”, there you may make changes, then click “ok”. It is modified.

CONCLUSION

With experience gained as a SAS programmer, in my estimate, sixty percent of the coding can be set up in a form of automation by creating macro buttons, examples were shown above, on SAS EG. I encourage you to take some time and explore the possibilities to pursue on this button-driven process. From my own experience, the efficiency gained from this semi-automation has raised my productivity. I have noticed more “free-time” that has been gained. Even for efficacy tables, the major chunk of code can be reused via these buttons. Hope the above concepts and ideas can assist you to expend your coding methods and save more time to analyze data, discuss complicated concepts with the clinical team, to do a clinical trial in an more efficient way.

ACKNOWLEDGEMENTS

The author would like to acknowledge Musa Nsereko and Amy Liang for their encouragement to pursue this concept.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. You may contact:

Charles Ling

Email: cling@incyte.com