Concomitant Medication Duration Calculation Using SAS® DS2 Procedure

Yirong Cao, Bristol Myers Squibb, Uxbridge, U.K.

Sai Jandhyala, Bristol Myers Squibb, Uxbridge, U.K.

ABSTRACT

Concomitant medications are medications other than investigational study drug which are taken at any time on-treatment. In practice, the calculation of overall duration can cause a lot of challenges due to the combination of complex scenarios including sequential treatment, on and off treatment and overlapping among these treatments. This paper solves these problems by using three different methods from the SAS DS2 procedure. All are available in SAS BASE and demonstrate enhanced programming efficiency: arrays, multi-threading, and packages. The DS2 procedure is the method of choice because of its advantages in re-usability, portability in SAS Viya, multithreading, numerical precision, and viability from multiple databases. Through the comparison with a simulated CM dataset, all three methods accurately calculate the overall duration of concomitant medication and highlight the different advantages: DS2 arrays are easy to be executed, multi-threading performs the best in terms of time efficiency, and DS2 packages can calculate all populations in one procedure.

Keywords: Concomitant medication, DS2 array, multi-threading, DS2 package

I. INTRODUCTION

During a clinical trial, a subject may take multiple concomitant medications (CM thereafter). The duration of CM has been requested by Health Technology Authority (HTA) to calculate the costs and resource burden. This can also facilitate the drug surveillance and capture the subjects' medication use. Typical scenarios include sequential treatment, on and off treatment and overlapping among these treatments. This causes lots of complexity when performing the overall duration calculation. There is a way to remove this obstacle by using a new method called the DS2 procedure. The benefits of this procedure include re-usability and portability, multithreading, numerical precision, and viability from multiple databases. This paper specifically utilises DS2 arrays, multithreads and DS2 packages. Any of these methods can be adopted when generating the CM duration and obtain the same results. However, they demonstrate advantages in different aspects: DS2 arrays are easy to understand and can be executed in one step, multithreading method exhibits the best time efficiency and additionally, DS2 packages can accommodate multiple populations in one procedure.

The following paper is organised as below: Section II introduces the background of the concomitant medication using a simulated dataset and explains the SAS DS2 procedure referencing three specific methods; Section III illustrates the duration calculation (using arrays,

multi-threading, and package functions) and compares the time efficiency across these different methods; Section IV draws a conclusion and discusses future research.

II. BACKGROUND

2.1 CM DATASET AND SIMULATION

Concomitant medications are either initiated before treatment starting date and continue after treatment starting date or initiated on/after treatment starting date. The CM dataset collects commitment/prior medications data in either a list type free text format (requiring coding by data management) or a pre-categorised data format. This data can be used in safety or efficacy analyses and the presence of specific medications may also be used as covariates for inferential analyses. For concomitant medication duration calculation, a list of medications that take effect on treatment is normally provided by the clinicians and may vary in different protocols.

The following scenarios are typically present in CM data:

- Sequential treatments: where each concomitant medication for a patient starts and stops sequentially within the timeline (without any overlap).
- On/off treatments: this refers to the same medication can be taken for a certain time and stops for a while, and then continues for another session. The interval between the end of the first medication occurrence and start of the next occurrence is considered as 'off treatment' and should therefore be discounted from the overall duration.
- Overlapping treatments: an overlap can exist between two or more concomitant medications where a patient takes and subsequently the number of overlapping days needs to be discounted from the total maximum duration calculation.

In this study, a simulated CM dataset is generated to reflect the above scenarios. It contains 100 subjects, which are equally assigned as treatment and control arm. Each subject has four concomitant medications (i.e., 1 – 4), and two of them (CM 1 and CM2) have on/off treatment such as 1 (before the 'off treatment' interval) and 1.1 (after the 'off treatment' interval). The CM start dates (cmstdt) are randomly assigned between 01 January 2001 and 21 June 2001, and the end dates (cmendt) follows a random number of days. The CM start day (cmstdy) and end day (cmendy) are calculated as start dates and end dates minus reference date (refdt) plus 1 respectively.

	usubjid	trt	trtn	cmstdt	cmendt	refdt	cmstdy	cmendy	cmseq
1	1	Treatment	1	16MAR2001	30MAR2001	27DEC2000	80	94	1
2	1	Treatment	1	21APR2001	28APR2001	27DEC2000	116	123	1.1
3	1	Treatment	1	28MAR2001	30MAR2001	27DEC2000	92	94	2
4	1	Treatment	1	08APR2001	11APR2001	27DEC2000	103	106	2.1
5	1	Treatment	1	17APR2001	18APR2001	27DEC2000	112	113	3
6	1	Treatment	1	13MAY2001	27MAY2001	27DEC2000	138	152	4
7	2	Control	2	15FEB2001	02MAR2001	27DEC2000	51	66	1
8	2	Control	2	12MAR2001	24MAR2001	27DEC2000	76	88	1.1
9	2	Control	2	27FEB2001	01MAR2001	27DEC2000	63	65	2
10	2	Control	2	31MAR2001	07APR2001	27DEC2000	95	102	2.1
11	2	Control	2	05MAR2001	08MAR2001	27DEC2000	69	72	3
12	2	Control	2	17MAR2001	28MAR2001	27DEC2000	81	92	4
13	3	Treatment	1	06JAN2001	23JAN2001	29DEC2000	9	26	1
14	3	Treatment	1	29JAN2001	04FEB2001	29DEC2000	32	38	1.1
15	3	Treatment	1	22JAN2001	31JAN2001	29DEC2000	25	34	2
16	3	Treatment	1	06FEB2001	20FEB2001	29DEC2000	40	54	2.1
17	3	Treatment	1	31JAN2001	12FEB2001	29DEC2000	34	46	3
18	3	Treatment	1	18MAR2001	25MAR2001	29DEC2000	80	87	4

The following graph illustrates a scenario where a subject has 4 concomitant medication events. The subject initiates medicine (CM1) for a duration of 11 days, and then after 13 days of off treatment, the subject receives CM1 again for a duration of 6 days. Then, the subject takes the second medicine (CM2) with on and off treatment too. However, he/she has the medicine together with the first medicine for 8 days and continues to have CM2 separately after 26 days for another 8 days. The third medicine (CM3) starts after the first durations of CM1 and CM2 but has some overlap with the second session of CM1. Finally, the patient has another drug (CM4) after all other medicines cease.

Here is the overall duration calculation for each and total concomitant medication:

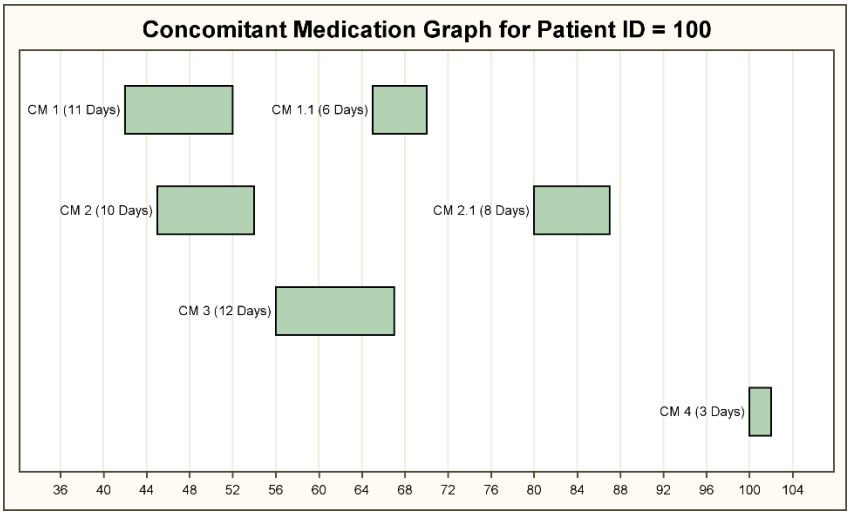
Duration for CM 1 = 11 (day 42 to day 52) + 6 (day 65 to day 70) = 17 days

Duration for CM 2 = 10 (day 45 to day 54) + 8 (day 80 to day 87) = 18 days

Duration for CM 3 = 12 days (day 56 to day 67)

Duration for CM 4 = 3 days (day 100 to day 102)

Total duration = 17 + 18 + 12 + 3 – 8 – 3 = 39 days



2.2 SAS DS2 PROCEDURE

The DS2 procedure is selected to calculate the duration because of the following key benefits:

a) Programmatic re-usability and portability is present as DS2 can be executed on SAS BASE as well as SAS Viya.
b) Multithreading capabilities enable the user to process multiple records or tasks in parallel. With multiple cores, significant performance gains can be achieved in comparison to traditional SAS data step methodology by pre-defining the number of threads to utilise.
c) Numerical precision can be enhanced as traditionally in data step methodology variables are either set to be numeric or character. However, other formats can be catered for in the DS2 procedure by declaring variables to have specific formats to avoid losing accuracy, e.g., DOUBLE and BIGINT.

d) Efficiency gains are potentially viable from multiple sources. It can be utilised for advanced resource intensive data manipulation.

This paper explains three elements of DS2 in details which are particularly applicable to concomitant medication duration analysis: arrays, multi-threading, and packages. The aim is to demonstrate multiple benefits to using DS2 procedure to determine the total duration of concomitant medication (accounting for on/off interval and overlapping events).

An array is a convenient way of homogeneous data collection for processing in the DS2 procedure. It has two types: temporary array and variable array. The first one can be generated by DECLARE statement, and the second one is through VARARRAY statement. The main difference between these types lies in that temporary array is a set of temporary elements, which can be in local or global scope. In comparison, variable array contains a set of references to variables in the Program Data Vector (PDV) and can be declared globally only. However, these two types of arrays have something in common too, e.g., the data must be homogeneous by type, multi-dimensional and indexed by the signed integer values, etc. Both arrays are adopted in this study due to different characteristics.

Multithreading facilitates the process of breaking up the input dataset and performing an operation on these split datasets then combining them back together. Every one of these operations is run in parallel, spread and processed across multiple cores on a single processor simultaneously. Each of these can be described as a thread. Computational time can be reduced when combing large tables from different data sources. However, if SAS system only has a single core CPU, the impact of multithreading can be negligible.

A DS2 package is designed for reusing the same procedures or variables for multiple times. It includes two types: pre-defined package and user-defined package. The former supports several functions such as FCMP, hash and hash iterator and SQLSTMT. The latter stores the methods that users create by themselves, which can be accessed with a DECLARE statement or the _NEW_ operator.

III. SAS DS2 Procedures

3.1 DS2 ARRAY

To calculate the duration, the most common way is to use arrays. DS2 has a similar way to solve this situation. The following method checks whether the CM Day is in any of the 6 concomitant medication sessions (the on/off medication is counted as 2 sessions) using the simulated dataset. If yes, then duration is accumulated.

The following example DS2 procedure embeds a data step and ends with ENDDATA statement. The variables are assigned as the array values using VARARRY statement, i.e., cmstart [6] and cmstop[6]. The other temporary array variables are created with the DECLARE statement and only used for this data step. There are multiple methods that can be selected in

DS2 procedure, and METHOD RUN() is selected here to link with the data step to run against each record. The SET statement calls a FEDSQL statement to select all records in the cm_horizon dataset. It can accommodate other databases besides SAS datasets.

The rest of code is like other procedures in SAS BASE. Each CM day (cmdy) is checked if it falls into any concomitant medication interval. If so, the variable init is set to be 1, and duration increases by 1. Otherwise, the duration remains the same. This process repeats from the first CM day to the last CM day, and the variable DUR will obtain the total duration of all medications.

```
proc ds2;
       data cm_cal (overwrite=yes);
              retain i dur;

              vararray double cmstart[6] cmstdy1 - cmstdy6;
              vararray double cmstop[6]  cmendy1 - cmendy6;

              declare double minstart maxstop cmdy i dur init;

              method run();

                 set {select * from cm_horizon} ;

                 minstart=min(of cmstdy:);
                 maxstop=max(of cmendy:);

                   dur=0;
                      do cmdy=minstart to maxstop;
                            i=1;
                            init=0;
                                   do while(init=0 and i<=6);
                                       if(cmstart[i]<=cmdy<=cmstop[i]) then init=1;
                                          i+1;
                                   end;
                            if init=1 then dur+1;
                      end;

               end;
       enddata;
   run;
quit;
```

3.2 ARRAY WITH THREADS

Further attempts have been performed using the multi-threading array method. This method has an impact on task turnaround time by determining the maximum number of threads for efficiency. The optimal number of threads is dependent on the number of CPU(s) factoring in the number of cores and logical processors available. If a thread is enclosed within a DS2 procedure, then data is dispersed across multiple processes for increased efficiency.

There are two procedures to process threads. In step 1, a new thread named NEWCAL is defined between the THREAD and ENDTHREAD statements. The rest of the code is similar to the above array method. However, this step does not include a data step.

```
proc ds2;
       thread newcal/overwrite=yes;
              retain i dur;
```

```
            vararray double cmstart[6] cmstdy1 - cmstdy6;
            vararray double cmstop[6]  cmendy1 - cmendy6;

            declare double minstart maxstop cmdy i dur init;

            method run();

               set cm_horizon;

               minstart=min(of cmstdy:);
               maxstop=max(of cmendy:);

                 dur=0;
                    do cmdy=minstart to maxstop;
                          i=1;
                          init=0;
                                 do while(init=0 and i<=6);
                                  if(cmstart[i]<=cmdy<=cmstop[i]) then init=1;
                                        i+1;
                                 end;
                          if init=1 then dur+1;
                    end;

             end;
      endthread;
run;
```

Step 2 is to process data with multi-threading. The thread (newcal) is invoked by the DECLARE statement. The input data is cm_horiszon with 10 threads as an example. The output dataset containing the total duration variable (dur) is subsequently generated called cm_cal_method2.

```
proc ds2;
      data cm_cal_method2 (overwrite=yes);
            declare thread newcal cm_horizon;
            method run();
                  set from cm_horizon threads=10;
            end;
      enddata;
run;
quit;
```

3.3 DS2 PACKAGE

There may be various ways to calculate the duration using DS2 packages. This paper provides one example on using the combination of SQLSTMT package and a user-defined package. The goal of this method is to convert the dataset to vertical format and then remove the duplicate days. The first step is to create a vertical dataset each CM day with the following code.

```
data cm_vertical;
      set cm_yc;
      do cmdy=cmstdy to cmendy;
         output;
      end;
run;
```

Take the first observation as an example: the CM start day (cmstdy = 80) and the CM end day (cmendy = 94) are expanded to 15 records for each CM day (cmdy). Based on the length of

CM duration, the whole simulated dataset contains 5735 records. Then, the total duration is calculated by removing all the duplicate CM day (cmdy) across all concomitant medication dataset.

	usubjid	cmstdy	cmendy	cmseq	cmdy
1	1	80	94	1	80
2	1	80	94	1	81
3	1	80	94	1	82
4	1	80	94	1	83
5	1	80	94	1	84
6	1	80	94	1	85
7	1	80	94	1	86
8	1	80	94	1	87
9	1	80	94	1	88
10	1	80	94	1	89
11	1	80	94	1	90
12	1	80	94	1	91
13	1	80	94	1	92
14	1	80	94	1	93
15	1	80	94	1	94

Step 2 creates a package called ‘cmv’. In this procedure, a pre-defined package ‘SQLSTMT’ is employed to create a FedSQL statement (stmt). This is sent to FedSQL language processor, which, in turn, sends the statement to DBMS to be prepared and stored. Two variables (tablename and indata) are customized to take different values when the packaged is declared in the next step.

```
proc ds2;
        package cmv /overwrite=yes;
             method output(char(100) tablename,
                           char(100) indata);

            declare package sqlstmt stmt('create table ' || tablename || ' as
                          select usubjid, count(distinct cmdy) as dur
                          from ' || indata ||
                          'group by usubjid');
                     stmt.execute();
             end;
        endpackage;
run;
quit;
```

The following code is used to declare the package ‘cmv’. The FedSQL statement (stmt) is executed in this step. This is realised by using f.output with assigning different values for ‘tablename’ and ‘indata’. For example, the first case takes the data from cm_vertical (indata), calculates the overall duration for all medicines and saves the results in cm_v1 (tablename). The second case specifies the data from cm_vertical for the first medicine (cmseq=1 or 1.1), and keeps the results in cm_v2, and so on and so forth.

```
proc ds2;
       data _null_;
          method init();
             declare package cmv f();
             f.output('cm_v1', 'cm_vertical');
             f.output('cm_v2', 'cm_vertical where cmseq=1 or cmseq=1.1');
             f.output('cm_v3', 'cm_vertical where cmseq=2 or cmseq=2.1');
             f.output('cm_v4', 'cm_vertical where cmseq=3');
             f.output('cm_v5', 'cm_vertical where cmseq=4');
          end;
```

```
        enddata;
run;
quit;
```

3.4 RESULTS

The above three methods generate a subject level dataset which is used to produce a descriptive statistics table that feeds into the HTA resource burden calculation. The results from each method are the same.

Descriptive Analysis for Concomitant Medication Duration

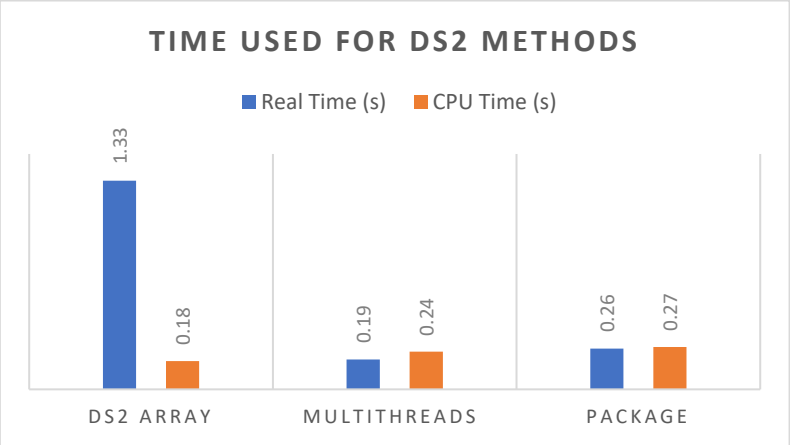
	N	Mean (95% CI)	s.d.
CM 1	100	17.8 (16.5, 19.1)	6.4
CM 2	100	19.4 (18.1, 20.8)	6.9
CM 3	100	8.3 (7.4, 9.1)	4.3
CM 4	100	11.8 (10.7, 13.0)	5.8
Total	100	47.1 (44.9, 49.2)	10.7

3.5 COMPARISON OF TIME EFFICIENCY

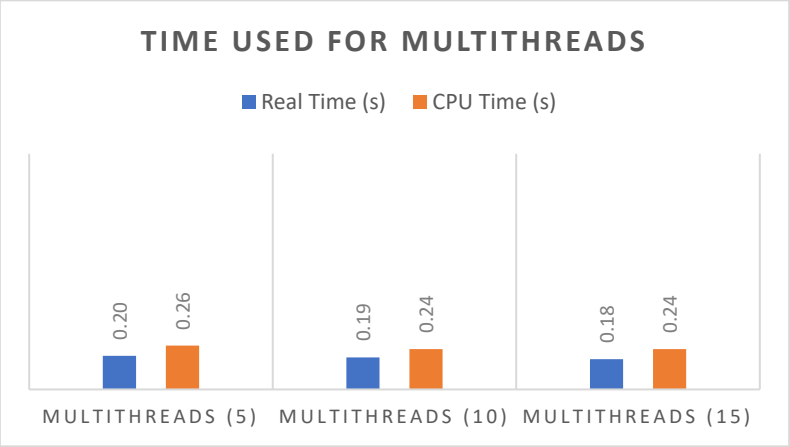
The machine used to conduct the testing has 4 cores and 8 processors. It runs PC-SAS 9.4 TS Level 1M6 on Windows 10 Pro-64bit.

CPU : Intel® Core™ i5-10310U @1.70GHz
Base Speed : 2.21GHz
Cores : 4
Logical Processors: 8

Below is the time efficiency comparison among the three methods. As the graph demonstrates, the real time for DS2 arrays is the most time consuming (1.33 seconds), compared to multi-threads and DS2 package methods. The CPU time does not differ too much. For the rest of two methods, multithreads perform the best even when the small sample size may shrink the comparison.



For multithreading, the impact of utilising an incremental number of threads is also examined. However, the difference observed from varying the number of threads between 5 and 15 is not significant.



Overall, multithreading method performs the best in both real time and CPU time among the three methods, and the variance of using different number of threads is invisible. It is worth noting that the processing time in SAS varies slightly every time, but the profile above remains the same.

IV. CONCLUSIONS

This paper has utilised three DS2 methods to calculate the complex duration calculation for concomitant medications, and effectively tackles the scenarios with the combination of sequential treatments, on-off treatments and overlapping. The results obtained are identical, but they show different advantages and disadvantages, e.g., multi-threading performs the best from the perspective of time efficiency although depends on the number of processors in the computer, and DS2 package is the most efficient in coding because it can accommodate multiple populations in one go.

Future research can expand the DS2 procedure to calculate the duration of Adverse Event which has all three scenarios as well. It can also call the data from other databases such as DB2 and MySQL and also work on the cloud based analytic platform, SAS Viya, to accelerate the processing time.

REFERENCES

Blum, J., J. Duggins. (2019) Getting Started with PROC DS2. SESUG, Paper 231-2019.

CDISC. SDTM Implementation Guide 3.2.

Cheng, A. (2012). Duration Calculation from a Clinical Programmer's Perspective. SUGI 31, Paper 048-31.

SAS Institute Inc. SAS 9.4 and SAS Viya 3.5 Programming Documentation: DS2 Procedure.

Shostak, J. (2005). SAS® Programming in the Pharmaceutical Industry. Cary, NC: SAS Institute Inc.

CONTACT INFORMATION:

Your comments and questions are valued and encouraged. Please contact:

Yirong Cao
Bristol Myers Squibb
Uxbridge
UB8 1DH
United Kingdom
Yirong.Cao@bms.com

Sai Jandhyala
Bristol Myers Squibb
Uxbridge
UB8 1DH
United Kingdom
Sai.Jandhyala@bms.com