PharmaSUG China 2015 - Paper 70

SAS® Longitudinal Data Techniques

- From Change from Baseline to Change from Previous Visits

Chao Wang, Fountain Medical Development, Inc., Nanjing, China

ABSTRACT

Longitudinal data is often collected in clinical trials to examine the effect of treatment on the disease process over time. The most common operation against this type data is to calculate the change from baseline (CFB). The proven ways to implement this calculation in SAS may include data splitting, one pass DOW (Dorfman-Whitlock DO) - Loop, retain plus LAG function. Sometimes, in addition to CFB, change from previous visits (CFP) could be interesting endpoints for a trial. This paper demonstrates how CFP can be derived with minimum coding by using SAS LAG (N) Function.

INTRODUCTION

In clinical trials, most of the endpoints are collected with multiple visits. It would be very helpful to examine the effect of treatment on the disease process over the time. And based on pharmaceutical industry practice, the collected data will be remapped to the SDTM and AdaM sequentially. This type data contains one or more records per subject, per analysis parameter, per analysis timepoint. We call it Basic Data Structure (BDS) in CDISC, and Longitudinal data in general. Figure 1 shows a general sample data.

	Subject Identifier for the Study	Analysis Relative Day	Parameter Code	Analysis Value	Baseline Record Flag
1	100001	-20	SYSBP	106	
2	100001	-1	SYSBP	105	Y
3	100001	1	SYSBP	106	
4	100001	2	SYSBP	106	
5	100001	3	SYSBP	118	
6	100001	4	SYSBP	115	
7	100001	13	SYSBP	116	

Figure 1 Sample Data

Change From Baseline (hereinafter referred to as "CFB") is a very common operation against this type data. There are many ways and techniques have been discussed in previous papers. In summary, there are 3 ways,

- Data Splitting: (Baseline part plus post-baseline parts)
- Retain Method
- One pass DOW (Dorfman-Whitlock DO) – Loop

But sometimes, calculate the Change FROM PREVIOUS VISITS (hereinafter referred to as "CFP") are also required. It's an interesting endpoint for some trials. Could the three CFB ways be expand used to calculate the CFP? This topic will be discussed in following part.

THREE METHODS OF CFB

The sample data will be used to show the ways to calculate the CFB. The variable names of the 5 columns are SUBJID, ADY, PARAMCD, AVAL and ABLFL and the dataset have been sorted by SUBJID, PARAMCD, ADY.

DATA SPLITTING

This method splits baseline data from the whole dataset. Then merge back by the key variables. It's a very simple way to get the baseline results. The example code is listed below,

```
data base post;
  set sample;
  if ABLFL = "Y";
run;
```

```
data _cfb1;
  merge sample(in=a) baseline(keep=subjid paramcd aval rename = (aval=base));
  by subjid paramcd;
  CHG = AVAL - BASE;
run;
```

RETAIN METHOD

Retain Statement is popular used in CFB calculation. It can carry the value of the specified variable to next iterations. The example code is listed below,

```
data _cfb2;
  set sample;
  by subjid paramcd ady;
  retain tmp 0 tmpdy 0;
  if ABLFL = "Y" then do;
    base = aval;
       tmp  = base;
       tmpdy= ady;
  end;

  if ady >= tmpdy then do;
    base = tmp;
    chg  = aval - tmp;
  end;
  drop tmp tmpdy;
run;
```

ONE PASS DOW LOOP

The DOW-Loop is a technique originally developed by Don Henderson, and popularized on the SAS-L listserv by Paul Dorfman and Ian Whitlock. The rational of this technique is taking control of the implicit DO-Loop inherent in the DATA step, identifying and storing the baseline value in a variable that is retained until all post-baseline values for a subject have been processed, and then writing out ***only*** those post-baseline values. The example code is listed below,

```
data _cfb3;
  do until (last.paramcd);
    set sample;
    by subjid paramcd ady;
      if ABLFL = "Y" then base = aval;
      else do;
        chg = aval - base;
        output;
      end;
  end;
run;
```

Compared the 3 outputs in Figure 2, we found the limitations of the 3 methods. The methods 1 and 3 are easy to calculate CFB for the post-baseline records. When we try to finish the calculation in one step, the RETAIN statement have more flexibly strength.

Data Splitting

	SUBJID	ADY	PARAMCD	AVAL	ABLFL	base	CHG
1	100001	-20	SYSBP	106		105	1
2	100001	-1	SYSBP	105	Y	105	0
3	100001	1	SYSBP	106		105	1
4	100001	2	SYSBP	106		105	1
5	100001	3	SYSBP	118		105	13
6	100001	4	SYSBP	115		105	10
7	100001	13	SYSBP	116		105	11

Retain Method

	SUBJID	ADY	PARAMCD	AVAL	ABLFL	base	chg
1	100001	-20	SYSBP	106			
2	100001	-1	SYSBP	105	Y	105	0
3	100001	1	SYSBP	106		105	1
4	100001	2	SYSBP	106		105	1
5	100001	3	SYSBP	118		105	13
6	100001	4	SYSBP	115		105	10
7	100001	13	SYSBP	116		105	11

ONE PASS DOW LOOP

	SUBJID	ADY	PARAMCD	AVAL	ABLFL	base	chg
1	100001	-20	SYSBP	106			
2	100001	1	SYSBP	106		105	1
3	100001	2	SYSBP	106		105	1
4	100001	3	SYSBP	118		105	13
5	100001	4	SYSBP	115		105	10
6	100001	13	SYSBP	116		105	11

Figure 2 The outputs of 3 CFB methods

THE METHODS TO CALCULATE CFP

When facing the CFP, many weapons present to our eyes - Array, Do Loop, Retain, LAG, DIF and etc. And the three CFB methods may expand to solve this question. Talk more details, a subject may have 5 visits (V1, V2, V3, V4 and V5). So we need to calculate the V_{21} (namely V2 change from V1), V_{31}, V_{41}, V_{51}; V_{32}, V_{42}, V_{52}; V_{43}, V_{53}; and V_{54}.

More generally, a subject may have N visits, so we need to calculate V_{ji} (j = 2…N, I = 1…j), the total number of V_{ji} is N*(N-1)/2. Wow, the problem become a little more complicated, especially the number of VISITs are not identical for different subjects. It will be very hard to apply the **Data Splitting**. The **One Pass DOW Loop** method is similar as **RETATIN method**, both of them use the iteration of Data Step. But the RETAIN method is more clarity.

A general way is using the ARRAY to transpose the visit information into horizontal, then use the DO-Loop to calculate the result by each interval. Is there any **longitudinal** ways? The answer is YES. In high school, we may use a trick to solve some hardly equation questions. Like, X – Y = (X – Z) + (Y-Z). So,

- V42 = V4 – V2 = V4 – V3 + V3 – V2
- V52 = V5 – V2 = V5 – V4 + V4 – V3 + V3 – V2

Then LAG/DIF function comes to our mind. We can summarize the DIF to get the result. Generally, the V_{ji} =

$$\sum_{k=i,i<j}^{j-1} DIF(k+1)$$

Bravo, we got the formula. Let use the two-levels do loop to solve this issue. One level is DO j from 2 to N, another level is DO from i from 1 to j-1. The speaking is easy, but it's very hard to make it come true. The problem spot is the sum (the SIGMA part). SAS only provides the horizontal summation function. This statement may be challenged by others. Because SAS also provides the SUM, a vertical summation tool, like RETAIN, a component to handle the longitudinal data. But the SUM tool is relied on RETAIN. It will make the programming work more complexity.

When searching the LAG/DIF function, we found these two functions have some special attributes. In SAS official document, the statement of LAG function is below,

LAG <*n*> (*argument*)

Unlike many other functions, it not only have argument, but also have a 'dimension'. The *n* specifies the number of lagged values. DIF has the same powerful function.

Based on this feature, we can update the formula to this,

- V42 = V4 – V2 = V4 – V3 + V3 – V2 = V43 + V32
- V52 = V5 – V2 = V5 – V3 + V3 – V2 = V53 + V32

So the general formula will be $V_{ji} = V_{j,i+1}+V_{i+1,i}$. In SAS, it will be $DIF_{j-i} = DIF_{J-(I+1)} + LAG_i$

The mind behind this formula is: calculating the interval = 1 (V_{21}, V_{32}, V_{43} and $V_{i+1,i}$) firstly, then calculating the interval = 2 (V_{31}, V_{42}, V_{53} and $V_{i+2,i}$) and so on.

Use the sample data to do a test, after running the code, we got the result as Figure 3,

	SUBJID	ADY	PARAMCD	AVAL	ABLFL	n	dif1	dif2	dif3	dif4	dif5	dif6
1	100001	-20	SYSBP	106		1						
2	100001	-1	SYSBP	105	Y	2	-1					
3	100001	1	SYSBP	106		3	1	0				
4	100001	2	SYSBP	106		4	0	1	0			
5	100001	3	SYSBP	118		5	12	12	13	12		
6	100001	4	SYSBP	115		6	-3	9	9	10	9	
7	100001	13	SYSBP	116		7	1	-2	10	10	11	10

Figure 3 The Example of CFP

```
data sample1;
  set sample;
  retain n 1;
  by subjid paramcd ady;
  dif0 = dif(aval);
  if first.subjid then dif0 = .;
  if first.subjid then n = 1;
    else n+1;
run;

%macro diff;
data dif;
  set sample1;
  by subjid paramcd ady;
  array dif[6] dif1-dif6;
  dif[1] = dif0;
  %do i = 2 %to 6;
  %let j = %eval(&i.-1);
  dif[&i] = dif[&i-1] + lag&j.(dif0);
  if N < &i then dif[&i] = .;
  %end;
  drop dif0;
run;
%mend diff;

%diff;
```

In above codes, we can update the dimension of ARRAY by using the macro variables, it will make the program more flexible. And we can also put the DIF1-DIF6 into one variable and made the dataset vertically. It will be more useful for the analysis.

CONCLUSION

LAG*n*, DIF*n* and Retain are The Three Musketeers to handle the longitudinal data. The built-in iteration function of LAG*n* and DIF*n* provides more flexibility to retrieve and manipulate the records vertically. With the help of ARRAY, the user could write more simplicity codes to conquer the difficulty tasks.

REFERENCES

- Nancy Brucken, 2009. “One-Step Change from Baseline Calculations”, SAS Global Forum 2009. http://support.sas.com/resources/papers/proceedings09/081-2009.pdf
- Chakravarthy, Venky. 2005. “RETAIN or NOT? Is LAG Far Behind?”. PharmaSUG 2005 Conference Proceedings. http://www.lexjansen.com/pharmasug/2005/coderscorner/cc19.pdf
- H. Ament. 2011. “Learning to love the SAS LAG function”. PhUSE 2011. http://www.phusewiki.org/docs/2011%20Papers/CC08%20paperl.pdf

ACKNOWLEDGMENTS

The author would like to thank Shanshan Wang and Xin Ke, who generously provided helpful comments and encouragements.

CONTACT INFORMATION

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

Name: Chao Wang
Enterprise: Fountain Medical Development, Inc.
Address: Room 402, Building 43, No.70, Headquarter Base, Phoenix Road, Jiangning District
City, State ZIP: Nanjing, China
Work Phone: 025-86155007
E-mail: chao.wang@fountain-med.com