A Macro for Last Observation Carried Forward

Paper 2692-2018
Jonson Jiang, Syneos Health

ABSTRACT

This paper is about an application on the last observation carried forward (LOCF) method [1,2,3]. The last non-missing observed value is used to fill in missing values at a later time point when data is in longitudinal structure. There are two simple loops inside of Macro. The **k-loop** is for parameter variables and **i-loop** is for records continuously missing per parameter.

INTRODUCTION

In clinical trial, some patients could dropout some visits due to AE or other reasons. We may use the last observed non-missing value to fill in missing values at a later point. That is the Last Observation Carried Forward (LOCF) imputation method. The assumption for this imputation is the response remains constant at the last observed value. In general, we can use this method when data are in longitudinal structure. For example, repeated measures were taken per subject by time point.

In order to get previous value of measurements for a missing time-point, we can use 'lag( )' function in SAS. For example,

```
retain preval_var;

preval_var=lag(var);

if first.usubjid then preval_var='';
if var='' then do;
      if preval_var^='' then var=preval_var;
end;
```

There are two function parts in above codes. At first, to retain Pre-value Variable and set Pre-value Variable as lag of original variable. The second, to set the first subject's Pre-value Variable as missing and then check if value of current variable is missing and if Pre-value Variable is non-missing. If yes, to impute the value of Pre-value Variable as current record.

In clinical trial study, every patient could have difference continuously dropout visit and some could have more than one. And sometimes we have many parameter variables required to impute. To handle these situations, the Macro discussed below can be automatically functioned it well.

SAS CODES AND DISCCUSION

Let us discuss one parameter variable first.

```
%do i=1 %to 10; * i loop for continuously missing records *;

   data carry1;  set carry1; by &sort.;
       retain var_pre&i.;
       var&i.=&var.;
       var_pre&i.=lag(var&i.); * get previous value for certain parameter *;

          if first.usubjid then do; var_pre&i.='';  end;
          else do;
              if &var.='' and var_pre&i.^=' ' then do;
                 &var.=var_pre&i.;
                 &var._IMPFG='Y';
              end;
          end;

          keep &sort.  &var.  &var._ORG   &var._ IMPFG;
   run;
%end; * end do i *;
```

The “do loop”- i variable is for how many time-point data continuously missing. Let’s set to 10. It should cover all missing records’ situation. Based on the example logic we just discussed above, the “do loop”- i macro variable is suffixed as the name of previous value variable or current variable. In order to trace which record is imputed, an imputation flag named as ‘_IMPFG‘ is created.

The “Do loop - i = 1” is used to carry forward for the first missing record; If two records continuously missing, “Do loop - i=2” will be used; By analogy, number total of ‘i’ in “do look” will apply for number of continuously missing records.

Now we can discuss more than one parameter variables situation.

```
%macro run_carryover(datain=, dataout=, sort=, var1=);

  proc sort data=&datain.; by &sort.; run;

  data _null_; set &datain.;
      tot=put(count("&var1.", ' ') +1,  2.);
      call symput("tot", tot);
  run;

 %do k=1 %to &tot.; * k loop for each of parameter variable *;
     %let var=%scan (&var1., &k.); * scan to get each parameter variable *;

     data carry1; set &datain.; by &sort.;
       &var._ORG=&var.; * assigned xxx_ORG variable to keep original value *;
     run;

   %do i=1 %to 10; * i loop for continuously missing records *;
     ....;
   %end; * end do i *;
```

```
    %if &k=1 %then %do;
        data carry2; set carry1; by &sort.; run;
    %end;

     %else %do;
         data carry2; merge carry2 carry1; by &sort.; run;
     %end;
   %end; * end do k *;

   data &dataout.; *added all parameter variables into final output datasets*;
      merge carry2 &datain.(drop=&var1.);
      by &sort.;
   run;

%mend run_carryover;

%run_carryover (datain=test_in, dataout=test_out4, sort=%str(usubjid VISITNUM
LBDT), var1=%str(ALB HDL ALP CREAT));
```

For parameter variables in Macro call, we can assign as many as we need in the string of “var1” macro calling variable. For example, we can assign 4 parameter variables as ALB, HDL, ALP and CREAT.

Using COUNT and SYMPUT function to get total number of parameter variables as the end of “k - loop”. In “k – loop”, using “SCAN” function to pick up each parameter variable from ‘&Var1 - string’ variables in Macro call. Then extension name as “_ORG” variable was assigned to keep original value for this parameter variable.

As we discussed, we can see the “i - loop” is inside of “k – loop”. The “k - loop” picks up parameter variable and then enter into “i – loop” to use the last observed non-missing value to fill in missing values at a later point.

If we take a look “i - loop” codes in detail, we will find there is no ‘k’ stamp inside. That means “i - loop” variable will be replaced by every ‘k’ loop call. So we have to keep only &sort., &var., &var._ORG and &var._impfg at the end of “i – loop”; Otherwise when merging each “k – loop” call, it will be over-written by the same variable name.

At the last part of this Macro call, final imputing dataset is merged with call-in dataset to add all of other parameter variables which are not listed on &var1 calling variable. So the Macro output datasets will keep all variable in call-in dataset.

RESULTS

Running this Macro, the result was got as following.

Patient ID	VISIT Number	Test date	ALB	ALB_ORG	ALB_IMPFG	HDL	HDL_ORG	HDL_IMPFG	ALP	CL	CREAT	HBCAB	KETONE	LDL
001	1	7/4/12 8:00	35	35		1.01	1.01		99	101	124	NEGATIVE	0	2.31
001	2	7/24/12 8:00	35	35		1.01	1.01		98	106	121	NEGATIVE	0	2.31
001	3	8/4/12 8:00	32	32		1.01		Y	110	110	125			2.54
001	6	8/29/12 8:00	34	34		1.01		Y	109	108	132			2.54
001	7	9/2/12 8:00	34	34		1.09	1.09		109	108	125			2.54
001	7.01	9/2/12 14:00	34		Y	1.09	1.09		112	112	126		0	
001	8	9/18/12 0:00	35	35		1.09	1.09		112		148			2.69
001	9	9/30/12 0:00	35	35		1.09		Y						2.69
001	11	11/2/12 0:00	35	35		1.09		Y	92	117	126	NEGATIVE		2.69
001	15	12/4/13 8:00	36	36		1.09		Y	92	117	126		0	3.12
001	15.01	12/5/13 8:00	36		Y	1.07	1.07		86	112	127			3.32
001	15.02	12/6/13 8:00	36		Y	1.07	1.07		87	116	126	NEGATIVE		3.32
002	1	7/4/12 8:00	46	46		1.07	1.07		77	79	99		0	2.69
002	2	7/24/12 8:00	46	46		1.07		Y		78	98	NEGATIVE	0	2.69
002	3	8/4/12 8:00	46	46		1.02	1.02		76	79	97	NEGATIVE		2.87
002	3.01	8/4/12 13:00	46		Y	1.02	1.02		75	78	99			2.87
002	6	9/2/12 8:00	48	48		1.02		Y	77	72	93			2.87
002	6.01	9/2/12 14:00	48	48		1.02		Y	86	89	95			
002	8	9/18/12 0:00	49	49		1.02		Y	82	85	96			3.01
002	9	9/30/12 0:00	49		Y	1.02		Y	82	85	89		0	3.02
002	10	11/2/12 0:00	49		Y	1.05	1.05							3.21
002	12	12/4/13 8:00	48	48		1.05		Y		82	84	NEGATIVE		3.15
002	13.01	12/15/13 8:00	48	48		1.05	1.05		89	82	84			3.14
002	13.02	12/15/13 12:00	48		Y	1.08	1.08		89	82	84			
002	13.03	12/16/13 8:00	48		Y	1.08	1.08		89	82				3.21
002	13.04	12/17/13 8:00	48		Y	1.08	1.08		89		84			3.21

We can see there are two new variables for each parameter. For example, HDL_ORG and HDL_IMPFG new variables are for HDL parameter variable. Every HDL_ORG missing records per subject were imputed from the closest observed non-missing value into original HDL variable.

SUMMARY AND CONCLUSION

The last observation carried forward (LOCF) method is a common way for imputing data with dropouts in clinical trial study. The last non-missing observed value is used to fill in missing values at a later time point.

There are two simple loops inside of Macro. The **k-loop** is for parameter variables and **i-loop** is for records continuously missing per parameter.

This Macro also can be used if carry forward from the closest visit with non-missing value **after** the current time-point. We just change '**Sort**' Macro variable use "**descending**' order.

REFERENCES

(1) Last Observation Carried Forward Versus Mixed Models in the Analysis of Psychiatric Clinical Trials, Robert M. Hamer, et al, Am J Psychiatry 166:639-641, June 2009.

(2) Statistical Solutions, SOLAS, for Missing Data Analysis, 2008.

(3) LOCF Method and Application in Clinical Data Analysis, Huijuan Xu, NESUG 2009 Posters

CONTACT INFORMATION

Jonson Jiang

Principal Stat Programmer

Syneos Health

Tel: (425) 216-3809

E-mail: jonson.jiang@SyneosHealth.com

Web: www.syneoshealth.com