CC-003

A User-Friendly Macro for Last Observation Carried Forward

Tse-Hua Shih, Ph. D., inVentiv Clinical Solutions, LLC., Baltimore, MD

Yuan-Chi Chang, M.A., inVentiv Clinical Solutions, LLC., Baltimore, MD

Eugenia Henry, Ph. D., inVentiv Clinical Solutions, LLC., Baltimore, MD

Cliff Meng, Ph. D., inVentiv Clinical Solutions, LLC., Baltimore, MD

Xitao Fan, Ph. D., University of Virginia, Charlottesville, VA

ABSTRACT

When dealing with longitudinal data, biostatisticians frequently perform Last Observation Carried Forward (LOCF) before analyses. This paper provides a macro that instantly prepares LOCF for one or multiple variables. To use this macro, a user simply specifies number of visits and names of variables that needs LOCF. The current paper also provides alternative codes for different types of LOCF.

MOTIVATION

LOCF is a common procedure of imputation for longitudinal data in the pharmaceutical industry. It is so common that we don’t want to rewrite codes whenever it needs to be done again.

Code 1 (Vertical LOCF with Original Visits)

```
data raw;
input id vis a b c d e;
cards;
1 1 1 6 7 8 9
1 2 . 2 3 4 8
1 3 2 3 . . 5
2 1 4 5 . 5 .
2 3 7 5 . . .
2 4 6 7 3 5 3
3 2 9 4 5 7 9
3 3 . . . . .
3 4 7 . . . .
4 1 9 3 9 4 8
4 2 7 . . . .
4 3 . 6 . . 4
5 1 6 5 7 4 6
5 2 . 2 4 . 6
5 3 8 . 5 5 .
5 4 . 8 6 5 4
;
run;

%macro v_locf (score, in, out);

data a;
length x $ 100;
x = "&score"; output;
n=countw(x,' ');
do i=1 to n;
  value=scan(x,i,' ');
  call symputx('val'||trim(left(i)),value,'g');
  call symputx('total',n);
end;
run;
%put _user_;

proc sort data=&in out=x nodupkey ;  by id; run;
/*LOCF*/
data post;
do until ( last.id ) ;
  set raw ;
by id ;
%do k = 1 %to &total;
if &&val&k ^= . then l_&&val&k = &&val&k;
%end;
output ;
end ;
run ;
/* Get LOCF Flag*/
data &out;
  set post;
%do k = 1 %to &total;
if l_&&val&k ^= . and &&val&k = . then f_&&val&k = 1;
```

```
%end;
run;

%mend;
%v_locf (%str(a b c d e), raw, done);
/*%v_locf (variables that need LOCF, input dataset, output dataset);*/

proc print data=done;
run;
```

Code 2 (Vertical LOCF with Added Visits)

```
data raw;
input id vis a b c d e;
cards;
1 1 1 6 7 8 9
1 2 . 2 3 4 8
1 3 2 3 . . 5
2 1 4 5 . 5 .
2 3 7 5 . . .
2 4 6 7 3 5 3
3 2 9 4 5 7 9
3 3 . . . . .
3 4 7 . . . .
4 1 9 3 9 4 8
4 2 7 . . . .
4 3 . 6 . . 4
5 1 6 5 7 4 6
5 2 . 2 4 . 6
5 3 8 . 5 5 .
5 4 . 8 6 5 4
;
run;

%macro v_locf_add (score, visit, v_n, in, out);

data a;
length x $ 100;
x = "&score"; output;
n=countw(x,' ');
do i=1 to n;
  value=scan(x,i,' ');
  call symputx('val'||trim(left(i)),value,'g');
  call symputx('total',n);
end;
run;
%put _user_;

proc sort data=&in out=x nodupkey ;  by id; run;
/*Add missing visits*/
data dummy (keep = id &visit);
    set x;
      by id;
if first.id;
```

```
      do &visit = &v_n; output; end;
proc sort;
by id &visit;
run;

proc sort data=&in ;
by id &visit;
run;

data d;
   merge raw dummy;
by id &visit;
run;

/*LOCF*/
data post;
do until ( last.id ) ;
  set d ;
by id ;
%do k = 1 %to &total;
if &&val&k ^= . then l_&&val&k = &&val&k;
%end;
output ;
end ;
run ;
/* Get LOCF Flag*/
data done;
  set post;
%do k = 1 %to &total;
if l_&&val&k ^= . and &&val&k = . then f_&&val&k = 1;
%end;
run;

%mend;
%v_locf_add (%str(a b c d e), vis, %str(1,2,3,4), raw, done);
/*%v_locf_add (variables that need LOCF, name of visit variable,
sequential order of all possible visits, input dataset, output
dataset);*/

proc print data=done;
run;
```

Code 3 (Horizontal LOCF with the Last Existing Data Carried Forward)

```
data one;
 input a 1 b 3 c 5 d 7 e 9 f 11;
cards;
1 2   7 8
1 2 5 7 8
1 2 5   8 9
1   5 7 8 9
1 2
1
;
```

```
run;

%macro l_locf_last (in, out, score, num);

data &out (drop=i hold j) ;
  set &in;
/* put all variables into an array*/
array t(&num) &score ;
/* work backwards from the last variable to find the first non-missing
value */
do i=&num to 1 by -1;
/* Assign its value to a new variable 'hold' */
 if t(i) ne . then do;
    hold=t(i);
    leave;
 end;
end;
/* assign the value of 'hold' */
 do j=i+1 to &num;
   t(j)=hold;
 end;
run;

%mend;
%l_locf_last(one, two, %str(a b c d e f), 6);
/*%l_locf_last(input dataset, output dataset, variables that need LOCF,
number of variables that need LOCF);*/

proc print data=two;
run;
```

Code 4 (Horizontal LOCF Filling Missing Cells with Previous Values)

```
data one;
 input a 1 b 3 c 5 d 7 e 9 f 11;
cards;
1 2   7 8
1 2 5 7 8
1 2 5   8 9
1   5 7 8 9
1 2
1
;
run;

%macro l_locf_all (in, out, score, num);
data &out (drop=i hold j) ;
  set &in;
array t(&num) &score ;
do i=1 to &num ;
 if t(i) ne . then hold=t(i);
    if t(i)=. then t(i)=hold;
 end;
run;
```

```
%mend;
%l_locf_all (one, two, %str(a b c d e f), 6);
/*%l_locf_all(input dataset, output dataset, variables that need LOCF,
number of variables that need LOCF);*/

proc print data=two;
run;
```

CONCLUSION

By using basic data steps instead of the function “retain”, we provide example macros of LOCF both vertically and horizontally for different data situations so that readers can apply those techniques in their jobs.

CONTACT INFORMATION

Your comments and questions can be directed to:

Shih Tse-Hua (Jeremy)

Email: jshih@inventivclinical.com

TRADEMARK INFORMATION