

LOCF Programming with 6-Step Index Mapping

Homer Wang, Principal Analysis Programmer



Your Date Here

LOCF Index Mapping Programming 6-Steps

1 Create LOCF dataset with qualified LOCF feeding records

2 Define the desired time points

3 Identify the missing time points by index

4 Map in the missing time points by index

5 Define the structure of the created LOCF dataset

6 Define the variables to keep/drop

1. Create LOCF dataset with qualified LOCF feeding records

- ▶ Start from the finished dataset with all OC records and variables;
- ▶ Apply the conditions such as post baseline records prior to the last dosing date etc. (DO NOT apply ANL01FL='Y' as there might be multiple records within one visit window with later date/time);
- ▶ Take the very last record from the visit window prior to the missing time point.

1. Create LOCF dataset with qualified LOCF feeding records (code)

```
data locf1t; /* LOCF feeding dataset to feed AVAL to the LOCF records */  
  set locf1t; /* finished OC dataset with all vars derived */  
  by USUBJID AVISITN ADTM;  
  if last. AVISITN; /* record from the last available time point */  
    * Index to define which record to LOCF from;  
  locfavis = avisitn; /* Index of LOCF records*/  
run;
```

2. Define desired time points

- ▶ Select the desired time points (analysis visits) required by SAP;
- ▶ Create a dummy shell dataset with all the desired time points (analysis visits) in case there is missing time points from the LOCF feeding dataset;

2. Define desired time points (code)

```
data locfshels (drop=i j); /* Dummy shell dataset with all desired time points */
  length PARAMCD $8;
  set locfshels;
  by USUBJID;
    do j=1 to &paramct; /* Dummy all paramters */
      PARAMCD = put(j, paracd.);
    do i=&vis_list;
      AVISITN = i; /* Dummy all time point*/
      output;
    end;
  end;
run;
```

3. Identify the missing time points by index

- Transpose the LOCF feeding dataset with missing index to indicate the missing time point.

```
proc transpose data=locf1t out=tranlocf prefix=&prefix;  
  by USUBJID PARAMCD;  
  var locfavis; /* Index to indicate the LOCF feeding record*/  
  id AVISITN;  
run;
```

3. Identify the missing time points by index (cont)

Transposed dataset with missing index:

USUBJID	PARAMCD	NAME	WEEK2	WEEK4	WEEK8	WEEK12	WEEK14	WEEK16	WEEK20	WEEK24
GS-US-417-0302-12345	PHGADA	LOCFAVIS	2	4	8	12	14	16	20	24
GS-US-417-0302-12354	PHGADA	LOCFAVIS	2	4	8	12		16	20	24
GS-US-417-0302-13245	PHGADA	LOCFAVIS	2	4	8		14			
GS-US-417-0302-13254	PHGADA	LOCFAVIS	2	4	8	12	14			
GS-US-417-0302-14235	PHGADA	LOCFAVIS	2	4	8	12	14	16	20	24

4. Map in the missing time points by index

- ▶ Fill in the missing index with the time point index from the prior time point;
- ▶ Transpose the dataset with one record per time point. The missing time point will have the **index** value of the feeding LOCF time point;
- ▶ Merge the above dataset with the shell dataset by timepoint to create the **index** ready LOCF dataset.
- ▶ Merge the **index** ready LOCF dataset with the LOCF feeding dataset by **index** to fill in the missing LOCF values.

4. Map in the missing time points by index (code)

```
do i = 1 to (&avisitct - 1);  
    if locfd [i+1] eq . and locfd [i] ne . then  
        locfd [i+1] = locfd [i]; /* fill in the missing time points with index */  
    end;  
/* Create index ready LOCF dataset */  
proc transpose data=tranlocff out=backlocf;  
    by USUBJID PARAMCD;  
run;
```

4. Map in the missing time points by index (cont)

- Fill in the missing index with the time point index from the prior time point

USUBJID	PARAMCD	NAME	WEEK2	WEEK4	WEEK8	WEEK12	WEEK14	WEEK16	WEEK20	WEEK24
GS-US-417-0302-12345	PHGADA	LOCFAVIS	2	4	8	12	14	16	20	24
GS-US-417-0302-12354	PHGADA	LOCFAVIS	2	4	8	12		16	20	24
GS-US-417-0302-13245	PHGADA	LOCFAVIS	2	4	8		14			
GS-US-417-0302-13254	PHGADA	LOCFAVIS	2	4	8	12	14			
GS-US-417-0302-14235	PHGADA	LOCFAVIS	2	4	8	12	14	16	20	24
USUBJID	PARAMCD	NAME	WEEK2	WEEK4	WEEK8	WEEK12	WEEK14	WEEK16	WEEK20	WEEK24
GS-US-417-0302-12345	PHGADA	LOCFAVIS	2	4	8	12	14	16	20	24
GS-US-417-0302-12354	PHGADA	LOCFAVIS	2	4	8	12	12	16	20	24
GS-US-417-0302-13245	PHGADA	LOCFAVIS	2	4	8	8	14	14	14	14
GS-US-417-0302-13254	PHGADA	LOCFAVIS	2	4	8	12	14	14	14	14
GS-US-417-0302-14235	PHGADA	LOCFAVIS	2	4	8	12	14	16	20	24

4. Map in the missing time points by index (code)

```
data locfall1; /* index ready LOCF dataset */  
  merge locfshels (in=aa) backlocff (in=bb);  
  by USUBJID PARAMCD AVISITN;  
  if aa and bb;  
run;
```

/* Merge with the LOCF feeding data by index to fill in the missing LOCF values */

```
data locffilled;  
  merge locfall1 (in=aa) locf1t (drop=avisitn DTYPE);  
  by USUBJID PARAMCD locfavis;
```

4. Map in the missing time points by index (cont)

- Merge in the missing value by index

Index ready LOCF dataset:

USUBJID	DTYPE	PARAMCD	AVISITN	LOCFAVIS	AVAL
GS-US-417-0302-12345	LOCF	PHGADA	12	8	52
GS-US-417-0302-12354	LOCF	PHGADA	16	14	57
GS-US-417-0302-13245	LOCF	PHGADA	20	14	57
GS-US-417-0302-13254	LOCF	PHGADA	24	14	57
GS-US-417-0302-14235	LOCF	PHGADA	16	14	50
GS-US-417-0302-14253	LOCF	PHGADA	20	14	50
GS-US-417-0302-12345	LOCF	PHGADA	24	14	50
GS-US-417-0302-12354	LOCF	PHGADA	12	8	52

5. Define the structure of the created LOCF dataset

- Keep only the LOCF records.

`if AVISITN ne LOCFAVIS; /* Will keep only the LOCF records*/;`

- Create a complete LOCF set.

`set LOCF dataset with OC dataset.`

6. Define the variables to keep/drop

- ▶ Drop or set some variables on LOCF records to null. Such as VISIT/Date/Time etc.
- ▶ Set LOCF dataset with the OC dataset to get the final dataset.

LOCF Programming with 6-step Index Mapping

?