PhUSE EU Connect 2018

Paper PP24

First Aid Kit for an Observational Study

Yuliia Bahatska, Syneos Health, Berlin, Germany
Vladlen Ivanushkin, DataFocus, Berlin, Germany

ABSTRACT

Observational studies are not the kind of studies most of us come across every day and in many aspects, they are similar to the formal clinical trials however they have their own specifics. Of course, each observational study is different and has its own challenges but some of them are more likely to occur. The purpose of this paper is to highlight the most common challenges programmers experience while working on the observational studies and to suggest the means to deal with these challenges from the SAS® programming point as well as from the point of standards. Since observational study data does not necessarily need to be submitted to a regulatory agency such as the FDA there is no strict requirement for data standards. However, as ADaM has proved to be incredibly helpful when it comes to output creation here we will provide information about ADaM variables that might be used.

INTRODUCTION

In an observational study investigator observes subjects or measures certain outcomes without manipulation or intervention in contrast to randomized controlled trials where investigators do intervene and are interested in the effects of the intervention on an outcome. There are lots of different types of observational studies: cohort, case-control, cross-sectional; the studies can be prospective or retrospective. The data can be obtained in many various formats and usually the quality and consistency of the data is lower than for clinical trials. While for clinical trials CDISC standards provide strict and well-documented guidance, for observational studies everyone is on their own. Data collected in observational studies does not entirely fit in existing CDISC concepts: there are cases that can't be covered by ADaM and when it comes to SDTM there might be more violated rules than fulfilled. Therefore, the purpose of this paper is not only to provide programming advice and show challenging situations when ADaM variables can be utilized but also to encourage the programmers to collaborate in order to identify and address the challenges they face in their day-to-day life.

VISIT HANDLING

The CRF field "VISIT" that is common for clinical trials may often be not included in the CRF for observational studies. Or there might be no CRF at all. However, in most cases the endpoints are defined based on certain timepoints throughout the study and in such cases the values that were collected should be somehow assigned to these timepoints. Usually the date is still collected so this makes it possible to derive ADY, the study day. It is useful to keep in mind that the study day doesn't have to be based on treatment start date. According to ADaM implementation guide [1]: The reference date should be indicated in the variable-level metadata for ADY and the reference date should be included as a variable in the given analysis dataset or alternatively in ADSL. This can be particularly helpful in case there is no study drug. Also, for the timepoints of interest the programmers are usually able to define some kind of target day or target window. In ADaM terminology these would be stored in AWTARGET (Analysis Window Target) and/or AWLO (Analysis Window Beginning Timepoint) and AWHI (Analysis Window Ending Timepoint). Another variable that may come in handy during the programming is AWTDIFF, it represents the difference from target. After deriving the variables described above the programmers can proceed to visit mapping itself. Usually there is a clear rule in the SAP on how this should be done. For example, in case closest/first/last/worst/best value needs to be used the easiest way would be to set ANL01FL to "Y" on the respective value and use this flag for subsetting during the analysis programming. It may also happen that the average needs to be derived, for example, if two values are equally close to the target day. In this case the best way would be to create a new record and set DTYPE (derivation type) to "AVERAGE". In order to maintain data transparency, we strongly recommend not to delete any records from the dataset but rather use analysis flags and derivation type variable for subsetting. Although the data is not necessarily submitted it is always easier for both programmers and reviewers if the traceability is maintained. The table below illustrates how one can apply the AW variables.

Table 1 AW variables

Analysis Time Point (AVISIT)	Target Study Day (AWTARGET)	Time Window (AWRANGE)	Lower Limit Day (AWLO)	Upper Limit Day (AWHI)
Baseline	1	-		
1 week	8	day 2 – day 11	2	11
2 weeks	15	day 12 – day 18	12	18
3 weeks	22	day 19 – day 25	19	25
4 weeks	29	day 26 – day 32	26	32

Missing values imputation goes hand in hand with the visit mapping. It is usually done afterwards once it is clear which values need to be imputed. It may also happen that the values are not missing but due to some medical reason

specified in the SAP must be excluded from the analysis. ADaM implementation guide has a clear concept on what to do in such case: 'When an analysis timepoint is missing, the ADaM methodology is to create a new row in the analysis dataset to represent the missing timepoint and identify these imputed rows by populating the derivation type variable DTYPE: *'For example, when an LOCF/WOCF analysis is being performed, create LOCF/WOCF rows when the LOCF/WOCF analysis timepoints are missing, and identify these imputed rows by populating the derivation type variable DTYPE with values LOCF or WOCF. All of the original rows would have null values in DTYPE. It would be very simple to select the appropriate rows for analysis by selecting DTYPE = null for Data as Observed (DAO) analysis, DTYPE = null or LOCF for LOCF analysis, and DTYPE = null or WOCF for WOCF analysis.'*

To illustrate this situation let's look at the following example. There is a raw dataset with laboratory data without any visits. For better understanding SDTM-like variable names are used.

Table 2 Raw LB Dataset

Row	USUBJID	LBTEST	LBDT	LBORRES	LBORRESU
1	0001	Platelets	2017-02-02	305	(10^9/L)
2	0001	Platelets	2017-02-12	274	(10^9/L)
3	0001	Platelets	2017-02-16	300	(10^9/L)
4	0001	Platelets	2017-02-24	276	(10^9/L)
5	0001	Platelets	2017-02-25	274	(10^9/L)
6	0001	Platelets	2017-03-06	321	(10^9/L)
7	0001	Platelets	2017-03-10	325	(10^9/L)
8	0001	Platelets	2017-03-16	326	(10^9/L)
9	0001	Platelets	2017-03-19	290	(10^9/L)

The analysis is to be done for Baseline, Week 4, Week 8, Week 12, Week 16, Week 20 and Week 24 timepoints. All the ranges, target dates and rules to apply should usually be specified in the SAP.

Table 3 AVISIT mapping table from SAP

Row	AVISIT	AVISITN	AWTARGET	AWRANGE	AWLO	AWHI	AWU
1	Baseline	-1				1	DAYS
2	Week 4	4	29	2-43	2	43	DAYS
3	Week 8	8	57	44-71	44	71	DAYS
4	Week 12	12	85	72-99	72	99	DAYS
5	Week 16	16	113	100-127	100	127	DAYS
6	Week 20	20	141	128-155	128	155	DAYS
7	Week 24	24	169	156-183	156	183	DAYS

As there are no visits collected, information from Table 3 should be combined with the data of interest (in this case laboratory – Table 2). One of the ways is to perform many- to-many merge so that each observation in the data of interest would get the whole set of ranges. Then the AVISIT can be assigned. As the ranges should be not overlapping, only one visit value will be assigned to a record.
In this example laboratory values need to be assigned to visits using Table 3. For each visit values which fall into the corresponding range should be considered. If there are more than one value within the window, then the closest value to the target day should be chosen for analysis. In case there are multiple results equally close to the target day then average is to be derived and used for analysis. If no values fall into the window, the last observation is carried forward.
Also, it should be kept in mind that for the Baseline AVISIT in this case all values before the reference date (TRTSDT in this example) are to be considered and the VISITS dataset should be constructed correspondingly.

Table 4 AVISIT Mapping

Row	USUBJID	PARAM	AVISIT	AVISITN	LBDT	AVAL	TRTSDT	ADT	ADY
1	0001	Platelets (10^9/L)	Baseline	-1	2017-02-02	305	07FEB2017	02FEB2017	-5
2	0001	Platelets (10^9/L)	Week 4	4	2017-03-05	274	07FEB2017	05MAR2017	27
3	0001	Platelets (10^9/L)	Week 4	4	2017-03-09	300	07FEB2017	09MAR2017	31
4	0001	Platelets (10^9/L)	Week 4	4		287	07FEB2017		
5	0001	Platelets (10^9/L)	Week 8	8	2017-04-04	276	07FEB2017	04APR2017	57
6	0001	Platelets (10^9/L)	Week 12	12	2017-05-08	274	07FEB2017	08MAY2017	91
7	0001	Platelets (10^9/L)	Week 16	16	2017-05-23	321	07FEB2017	23MAY2017	106
8	0001	Platelets (10^9/L)	Week 16	16	2017-05-28	325	07FEB2017	28MAY2017	111
9	0001	Platelets (10^9/L)	Week 16	16	2017-06-07	326	07FEB2017	07JUN2017	121
10	0001	Platelets (10^9/L)	Week 20	20	2017-06-07	326	07FEB2017	07JUN2017	121
11	0001	Platelets (10^9/L)	Week 24	24	2017-06-07	326	07FEB2017	07JUN2017	121

Row	DTYPE	AWTARGET	AWTDIFF	AWLO	AWHI	ANL01FL
1 (cont)						Y
2 (cont)		29	2	2	43	
3 (cont)		29	2	2	43	
4 (cont)	AVERAGE	29		2	43	Y
5 (cont)		57	0	44	71	Y
6 (cont)		85	6	72	99	Y
7 (cont)		113	7	100	127	
8 (cont)		113	2	100	127	Y
9 (cont)		113	8	100	127	
10 (cont)	LOCF	141	20	128	155	Y
11 (cont)	LOCF	169	48	156	183	Y

In this case the traceability to raw data can be established via USUBJID, LBDT and LBTEST variables.
For AVISIT=Week 4 there are two values on study days 27 and 31 which are equally close to the target day. Therefore, an additional row is created with DTYPE=AVERAGE and ANL01FL is set to Y for this derived record. For AVISIT=Week 16 there are three values on study days 106, 111 and 121 but the one with ADT=28MAY2017 is closer to the AWTARGET and therefore marked with ANL01FL=Y. As there is no information after 07JUN2017, for both AVISIT=Week 20 and AVISIT=Week 24 the LOCF rule is applied and result from ADT=07JUN2017 is carried forward. LBTEST and LBDT are kept for these derived records to have clarity where they are taken from.

TREATMENT SECTION

One of the situations when observational study data does not fit into ADaM is the case when there is no study drug at all. According to the implementation guide ARM and TRT01P are required. However, as the data is not submitted to the FDA or any other regulatory agency it is not crucial to have the treatment section. When it comes to the derivation of study day other date than TRTSDT can be used as well, according to the CDISC note from ADaM IG v1.1 :

- *'The relative day of AVAL and/or AVALC. The number of days from an anchor date (not necessarily DM.RFSTDTC) to ADT.'*
- *'Note that it is possible to have different definitions for a relative day (or time) variable (e.g., ADY) in separate datasets, using different anchor dates (or times). For example, the derivation of ADY for efficacy datasets might be different from that for safety datasets.'*

Another suggestion would be to populate dummy values such as "No treatment" or "No drug".

DATE IMPUTATION

Working on observational studies usually involves quite a lot of date imputation. The information is often collected retrospectively; for many dates day and/or month part are missing, however this data must nevertheless be analyzed. In this section we will take a closer look at date imputation. From our experience worst case scenario is the best one when it comes to the date imputation. For example, if we have an adverse event that could have started before or after initialization of study drug, it is considered as the event that started after the treatment start date and therefore treatment-emergent. ADaM has a set of variables that are used in case date imputation is required. These are *DTF and *TMF variables: *'*DTF variables represent the level of imputation of the *DT variable based on the source variable. *DTF = Y if the entire date is imputed. *DTF = M if month and day are imputed. *DTF = D if only day is imputed. *DTF = null if *DT equals the respective variable date part equivalent. If a date was imputed, *DTF must be populated and is required. Both *DTF and *TMF may be needed to describe the level of imputation in *DTM if imputation was done.'* It is worth to mention the situation when the date is set to the last day of respective month. In this case it is not enough to just concatenate the respective value to the string with the partial date and use the INPUT function. Here we recommend using the INTNX function. Suppose we have a variable DATE_INCOMPLETE that stores only year and month in the format of YYYY-MM and we want to create a variable DATE_IMPUTED that would contain the date imputed to the last day of respective month. Then we could run the code below:

```
date_imputed=intnx('month',input(date_incomplete,anydtdte7.),0,'END');
```

The ANYDTDTE informat is used to create a SAS date from just the year and month. The INTNX function here adds 0 months to this date and END specifies that the returned date value is aligned to the end of the interval.

INTERIM ANALYSES

For observational studies, it's often required to perform interim analyses. For example, it may be envisaged in the SAP to do such analyses every year using cutoff dates. Cutoff of findings data shouldn't be a problem however it may be challenging for events and subject status at the cutoff date. For example, if event stop date is after the cutoff date, the event should be considered ongoing.
The cutoff date can be saved in ADSL in EOSDT and the subject status at the cutoff date - in EOSSTT. Similarly, the treatment status information may be placed into EOTSTT. To perform cutoff of the data it may be handy to put the cutoff date into a global macro-variable. This would allow just to change one macro-variable for following interim analyses. Also, additional endpoints may be applicable for the interim analysis.

SENSITIVITY ANALYSES

Similarly, to the clinical trials, analysis for observational studies also sometimes includes different kinds of subgroup and/or sensitivity analyses. The easiest and standard way to facilitate this is to create a set of needed variables in the ADSL and make them core variables. These variables would be merged to every analysis dataset and used for specified analyses.
The more efficiently you write your code, the less effort you will need to make a sensitivity or subgroup analysis. Using the COUNTW function can be helpful for macro parameters that might receive a list of values such as subgroups:

```
%do i=1 %to %sysfunc(countw(&subgroups));
   %local subgroup;
   /*get a value number &i from the subgroup parameter*/
   %let subgroup=%scan(&subgroups, &i);
   <And here you can include the code that needs to be repeated for the specific
   value of subgroup.>
    …
%end;
```

In case you have a subgroup parameter, which might contain more than one variable it is handy to use a regular expression to transform a space delimited value into a coma delimited one:

```
%let comma_delimited_subgroup=%sysfunc(prxchange(s/\s+/%str(,)/, -1 , &subgroup));
```

This is helpful, for example, in case the PROC SQL needs to be used.

CONCLUSION

While other areas of clinical research are well-standardized, working on observational studies still means reinventing the wheel in every single case. We hope that in this paper we have covered some of such cases and made programmers' life a little bit easier. Today observational studies are a subject of great interest to both programmers and CDISC. CDISC plans to release a white paper about data standards in observational studies, and in parallel PhUSE working group Optimizing the Use of Data Standards has started working on a similar project. In close cooperation they hope to provide guidance on how data for observational studies should be handled. If you would like to contribute you are more than welcome to contact the authors of this paper.

REFERENCES

[1] CDISC Analysis Data Model Team. "Analysis Data Model (ADaM) Implementation Guide". February 2016. http://www.cdisc.org/adam

CONTACT INFORMATION

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

Yuliia Bahatska
Syneos Health
Joachimstaler Str. 10–12,
10719 / Berlin, Germany
Email: yuliia.bahatska@syneoshealth.com

Vladlen Ivanushkin
DataFocus GmbH
Lothringer Str. 23,
D-50677 Cologne
Email: vladlen.ivanushkin@datafocus.de