Paper TA06

It is Time to Get Answers: Handling Questionnaire Data in Respiratory Studies

Valeriia Oreshko, Intego Group LLC, Kyiv, Ukraine
Ihor Savostianov, Intego Group LLC, Kharkiv, Ukraine

ABSTRACT

Questionnaires are one of the most important tools for respiratory clinical studies to evaluate the health and quality of patients' lives. However, a large amount of questionnaire data collected in the form of a patient's diary causes difficulties in preparing it for analysis. In this paper, we provide a summary of our experience in calculating time-to-event (TTE) endpoints based on questionnaires utilized in respiratory studies. Unlike other TTE endpoints, where an event of interest occurs at a single time point, our typical TTE endpoint may require the maintenance of the event for a certain period. This paper provides a method to trace patients' answers to produce meaningful outputs. We also consider common pitfalls of TTE design and its implementation in terms of ADaM Basic Data Structure (BDS). The following problems will be considered: dealing with missing data, common TTE calculation, requirements to baseline values, and the difference between event and censor date description.

INTRODUCTION

Time-to-event endpoints are commonly used in clinical trials to evaluate the efficacy of treatment. Moreover, they frequently serve as a primary endpoint. For this reason, it is crucial to ensure that such endpoints are correctly derived and that no details are overlooked. These endpoints hold a special feature in influenza and COVID-19 studies, where for an event of interest to be considered met, an appropriate criterion should be maintained for a certain period.

To develop a clear understanding of time-to-event, let us briefly remind its basic notions. Time-to-event, also known as survival, data comprise three components - origin, survival time, and censoring. Origin is the starting point from which we begin counting the time. Survival time or failure time is the duration between the origin and the occurrence of the event under consideration. Censoring variable marks if the event happened or not. Usually, the censoring variable equals 0 if the event happened and 1 or more if it did not. The examples of origin can be randomization time or the start of treatment. For COVID studies, an event of interest is, as a rule, the time of alleviation of a single symptom or all symptoms.

Let us now introduce a model example that will be considered throughout the paper. Assume that subjects have to fill out a symptom diary twice a day - in the morning and the evening - for 28 days. Symptom score range is defined as 0 – None, 1 – Mild, 2 – Moderate, 3 – Severe. *The time taken to alleviate a symptom is defined as the time between the treatment initiation and the start of a period in which symptom score <=1 and remains <=1 for 21.5 hours.*

Covid-19 Symptom Diary

Please rate the severity of each symptom at its worst using the following scale:

0 – None
1 – Mild
2 – Moderate
3 – Severe

№	Symptom	Day 1 Date: _____		Day 2 Date: _____	
		Morning Time: _____	Evening Time: _____	Morning Time: _____	Evening Time: _____
1	Headache				
2	Fainting/dizziness				
3	Fatigue				
4	Sore throat				
5	Shortness of breath				
6	Muscle Aches				
7	Vomiting				
8	Runny nose				
9	Chills/night sweats				
10	Cough				
11	Unable to care for self				
12	Little or no urination				
13	Loss of smell				
14	Loss of taste				

Figure 1.1 - Example of Patient's Covid-19 Symptom Diary

1. ADAM BASIC DATA STRUCTURE OF TTE DATASET

In general, ADAM time-to-event dataset has a similar structure to the usual BDS dataset, but with additional variables that support survival analysis. For this reason, in this paper we will only focus on the variables that give an account of time-to-event and will ignore other typical variables. The relevant variables and their descriptions can be found in Table 2.1. As a rule, TTE dataset .contains one record per subject, per parameter.

Variable Name	Variable Label	Core	Description
ADTM	Analysis Date/Time	Perm	Analysis of date/time of event or censoring associated with AVAL in numeric format.
STARTDTM	Time to Event Origin Date/Time	Perm	This is generally the time at which a subject is first at risk for the event of interest evaluation (as defined in the Protocol or Statistical Analysis Plan). For example, this may be the randomization date or the date of the first study therapy exposure.
AVAL	Analysis Value	Req	AVAL is the elapsed time to the event of interest from the origin based on ADTM and STARTDTM.
CNSR	Censor	Cond	CNSR displays whether the event has occurred or not. Usually, CSNR = 0 for the event and CNSR = 1 or > 0 otherwise.
EVNTDESC	Event or Censoring Description	Perm	Describes the event of interest or an event that warrants censoring.
CNSDTDSC	Censor Date Description	Perm	Describes the *date* that was used for censoring

Table 2.1 - Main ADTTE variables

The other variables like

- Identifier variables (STUDYID, USUBJID)
- Treatment variables (TRTA, TRTP)
- Parameter variables (PARAMCD, PARAM, PARCATy)
- Date/time imputation flag variables (--DTF, --TMF)
- Source variables (SRCDOM, SRCVAR, SRCSEQ)

should be included as well.

2. COMMON PITFALLS

ORIGIN OF THE EVENT AND BASELINE VALUE

One common mistake that programmers make is calculating time-to-event without considering the relative time of the event. This usually happens when they use all the available data without proper scrutiny. As a result, the event can occur before the treatment starts, as evidenced in Data Extract 3.1. Therefore, it will lead to negative time-to-event values that do not make sense as it can be only positive. Moreover, time-to-event endpoints are an integral part of *efficacy* analysis, and our ultimate goal is to evaluate the effect of the treatment. In other words, we want to know how a treatment under study alleviates the symptoms. Therefore, it is best to exclude subjects who have no or mild symptoms at baseline. With such reasoning in hand, we have come to the solution to our problem. (see Data Extract 3.2). By doing so, we can ensure that the event will never occur before the origin (start of treatment) since it should be maintained with no or mild score for a certain period.

PARAMCD	AVISIT	ATPT	AVAL
QS00101	Baseline	Morning	1
QS00101	Day 1	Evening	1
QS00101	Day 2	Morning	1
QS00101	Day 2	Evening	1
QS00101	Day 3	Morning	1

Data Extract 3.1 - Event starts at baseline before the treatment

PARAMCD	AVISIT	ATPT	AVAL
QS00101	Baseline	Morning	2
QS00101	Day 1	Evening	1
QS00101	Day 2	Morning	1
QS00101	Day 2	Evening	1
QS00101	Day 3	Morning	1

Data Extract 3.2 - Event starts after the treatment (baseline score is 2 - Mild)

CENSORING

If an event does not occur, it is called censoring. At first glance, it may seem to set the date to when the last symptom data was collected. Nevertheless, here is the catch. COVID-19 Symptom Diary cores are usually collected for a planned period being set in advance. However, a clinical trial does not always go according to the plan and data can be collected during unscheduled or follow-up visits after the period of our interest. It may lead to issues such as event time or maintenance of the event exceeding the preset time span, as seen in Data Extracts 3.4 and 3.5 (marked in red). This is something that we need to avoid, as the TTE endpoint is a crucial part of *efficacy* analysis, and we are interested in resolving symptoms within a certain time. Therefore, the proper approach is as follows: firstly, only consider event time and maintenance of the event within the period that was set in advance according to the protocol. Secondly, when an endpoint is not met, set the failure time as the end of the mentioned period (marked in blue in Data Extracts 3.3, 3.4, and 3.5).

PARAMCD	AVISIT	ATPT	AVAL
QS00101	Day 28	Morning	2
QS00101	Day 28	Evening	2
QS00101	UV 1	Morning	2
QS00101	UV 1	Evening	2
QS00101	UV 2	Morning	2

Data Extract 3.3 - Event has not occurred.

PARAMCD	AVISIT	ATPT	AVAL
QS00101	Day 28	Morning	2
QS00101	Day 28	Evening	2
QS00101	UV 1	Morning	1
QS00101	UV 1	Evening	0
QS00101	UV 2	Morning	0

Data Extract 3.4 - Event time exceeds time of data collection.

PARAMCD	AVISIT	ATPT	AVAL
QS00101	Day 28	Morning	2
QS00101	Day 28	Evening	1
QS00101	UV 1	Morning	0
QS00101	UV 1	Evening	0

Data Extract 3.5 - Maintenance of the event exceeds the time of data collection.

3. DEALING WITH MISSING DATA

Before starting to derive TTE endpoints, our goal is to obtain filled diaries for each planned time point. This means having twice-daily entries for 28 consecutive days. Otherwise, if any data is missing, we cannot confidently determine if the required criteria are still being met.

There are two types of missing data that can occur in the diary (Figure 4.1):

- Partially missing (at least one response is present for a time point)
- Fully missing (all responses are absent)

№	Symptom	Day 1 Date: 24-02-2022		Day 2 Date: 25-02-2022	
		Morning Time: 09:30	Evening Time: 18:30	Morning Time:	Evening Time: 19:00
1	Headache	1	1		1
2	Fainting/dizziness	1	1		1
3	Fatigue	1	1		1
4	Sore throat	2			2
5	Shortness of breath	0	1		0
6	Muscle Aches	1	2		1
7	Vomiting	1	0		1
8	Runny nose	1	1		1
9	Chills/night sweats	1	2		1
10	Cough	1			1
11	Unable to care for self	0	1		0
12	Little or no urination	0	1		0
13	Loss of smell	1	2		1
14	Loss of taste	1	3		1

Partially missing data: only symptoms 4 and 10 are absent

Fully missing data

Figure 4.1 – Examples of missing data in the diary

HANDLING OF PARTIALLY MISSING DATA

When a response to a question in a questionnaire is missing, the AVAL is set to the highest score on the scale, i.e. to the worst possible result, and will be treated further as a usually filled questionnaire. The records added to the dataset in such a way are marked with the variable DTYPE=WC. An example of this is shown in Figure 4.2, where a record is added for question 8 that was not collected for Day 5 Morning. In this case, the time point of ADTM is populated from the corresponding time point of present responses.

Total rows: 13 Total columns: 6

	USUBJID	QSTESTCD	QSSTRESN	VISIT	QSDTC
1	XXX-0001	QUEST1	1	DAY 5	2021-05-06T08:56:08
2	XXX-0001	QUEST2	2	DAY 5	2021-05-06T08:56:08
3	XXX-0001	QUEST3	1	DAY 5	2021-05-06T08:56:08
4	XXX-0001	QUEST4	2	DAY 5	2021-05-06T08:56:08
5	XXX-0001	QUEST5	2	DAY 5	2021-05-06T08:56:08
6	XXX-0001	QUEST6	2	DAY 5	2021-05-06T08:56:08
7	XXX-0001	QUEST7	1	DAY 5	2021-05-06T08:56:08
8	XXX-0001	QUEST9	2	DAY 5	2021-05-06T08:56:08
9	XXX-0001	QUEST10	1	DAY 5	2021-05-06T08:56:08
10	XXX-0001	QUEST11	1	DAY 5	2021-05-06T08:56:08
11	XXX-0001	QUEST12	1	DAY 5	2021-05-06T08:56:08
12	XXX-0001	QUEST13	0	DAY 5	2021-05-06T08:56:08
13	XXX-0001	QUEST14	0	DAY 5	2021-05-06T08:56:08

Total rows: 14 Total columns: 7

	PARAMCD	AVAL	ADTM	ATPT	AVISIT	DTYPE
1	QS00101	1	06MAY2021:08:56:08	MORNING	DAY 5	
2	QS00102	2	06MAY2021:08:56:08	MORNING	DAY 5	
3	QS00103	1	06MAY2021:08:56:08	MORNING	DAY 5	
4	QS00104	2	06MAY2021:08:56:08	MORNING	DAY 5	
5	QS00105	2	06MAY2021:08:56:08	MORNING	DAY 5	
6	QS00106	2	06MAY2021:08:56:08	MORNING	DAY 5	
7	QS00107	1	06MAY2021:08:56:08	MORNING	DAY 5	
8	QS00108	3	06MAY2021:08:56:08	MORNING	DAY 5	WC
9	QS00109	2	06MAY2021:08:56:08	MORNING	DAY 5	
10	QS00110	1	06MAY2021:08:56:08	MORNING	DAY 5	
11	QS00111	1	06MAY2021:08:56:08	MORNING	DAY 5	
12	QS00112	1	06MAY2021:08:56:08	MORNING	DAY 5	
13	QS00113	0	06MAY2021:08:56:08	MORNING	DAY 5	
14	QS00114	0	06MAY2021:08:56:08	MORNING	DAY 5	

Figure 4.2 – Example of imputed WC record

HANDLING OF FULLY MISSING DATA

The imputation of AVAL for fully missing assessments is based on “taking the next observation backward” considering the number of missed responses in a row. The following rules have been implemented:

1. If ***one or two consecutive*** entire symptom scores are missing, AVAL takes the result of the next non-missing assessment.
2. If ***more than two consecutive*** entire symptom scores are missing, AVAL considers that visit as having failed the endpoint.

These missing assessments are marked in a dataset with DTYPE=PHANTOM. For illustration purposes, we have included the examples of fully missing assessments before (Figure 4.3) and after (Figure 4.4) the imputation transposed by time point.

USUBJID	AVISIT	ATPT	QS00101	QS00102	QS00103	QS00104	QS00105
XXX-0001	DAY 10	MORNING	2	1	1	2	2
XXX-0001	DAY 10	EVENING	2	2	2	3	2
XXX-0001	DAY 11	MORNING	.	.	.	.	.
XXX-0001	DAY 11	EVENING	2	1	1	2	2
XXX-0001	DAY 12	MORNING	.	.	.	.	.
XXX-0001	DAY 12	EVENING	.	.	.	.	.
XXX-0001	DAY 13	MORNING	.	.	.	.	.
XXX-0001	DAY 13	EVENING	.	.	.	.	.
XXX-0001	DAY 14	MORNING	2	1	1	2	2
XXX-0001	DAY 14	EVENING	2	2	1	2	3
XXX-0001	DAY 15	EVENING	2	0	1	2	3
XXX-0001	DAY 16	EVENING	1	1	1	3	3
XXX-0001	DAY 17	EVENING	2	1	0	3	3
XXX-0001	DAY 18	EVENING	.	.	.	.	.
XXX-0001	DAY 19	EVENING	.	.	.	.	.
XXX-0001	DAY 20	EVENING	1	2	1	3	3
XXX-0001	DAY 21	EVENING	2	1	2	2	2

Figure 4.3 – Example of fully missing data (only 5 of 14 questions are shown)

Cases 1 and 3 fall under the first rule, which means that we populate AVAL from the next present assessment. In case 2, where 4 consecutive assessments are missed, we need to mark the endpoint as failed. To do this, we map these missed assessments as “99”.
To implement the logic described above, the first step is to create an auxiliary variable N_CONS_MISS. This variable indicates how missing records are present in a sequence. For the records from Figure 4.3 N_CONS_MISS would be the following:
Case 1: N_CONS_MISS=1, Case 2: N_CONS_MISS=4 Case 3: N_CONS_MISS=2. The next step is to execute the macro %TAKE_NEXT_RES (Code 4.4) to fill in the missing results with the next present one checking if the limitation for the maximum allowed number of the missing assessments is not exceeded. If there are two consecutive missing results (Case 3), the second execution repeats the procedure to populate the result. The result is populated from the second record of the missing sequence, which was filled during the previous macro execution.

```
*%let ren_lst=%str(qs00101=nextqs101 qs00102=nextqs102 qs00103=nextqs103 ...add all
questions);

%macro take_next_res(inds=, outds=);
   data &inds._2;
      *getting next values on the same line;
      merge &inds. &inds.(firstobs=2 rename=(&ren_lst.) keep=qs:);
   run;

   data &outds.(drop=next: i);
      set &inds._2;
      by usubjid avisitn avisit descending atpt adtm;
      if last.usubjid then call missing(of nextqs101-nextqs114);

      array quest [*] qs00101-qs00114;
      array next_quest [*] nextqs101-nextqs114;

      do i=1 to dim(quest);
         if missing(quest[i]) then do;
     *taking next value if there 1 or 2 consecutive missing values;
     if n_cons_miss<=2 and not missing(next_quest[i])
               then quest[i]=next_quest[i];

     *failed if there are more than 2 consecutive missing values;
     if n_cons_miss>2 then quest[i] = 99;
         end;
      end;
   run;
%mend take_next_res;
%take_next_res(inds=src_seq_miss, outds=_1_next_back);
```

```
%take_next_res(inds=_1_next_back, outds=_2_next_back);
```
Code 4.1 – Imputing missing results from the next observations.

The result after imputation data for the Case 1, 2 and 3 from Figure 4.3 you can see in Figure 4.4:

USUBJID	AVISIT	ATPT	QS00101	QS00102	QS00103	QS00104	QS00105
XXX-0001	DAY 10	MORNING	2	1	1	2	2
XXX-0001	DAY 10	EVENING	2	2	2	3	2
XXX-0001	DAY 11	MORNING	2	1	1	2	2
XXX-0001	DAY 11	EVENING	2	1	1	2	2
XXX-0001	DAY 12	MORNING	99	99	99	99	99
XXX-0001	DAY 12	EVENING	99	99	99	99	99
XXX-0001	DAY 13	MORNING	99	99	99	99	99
XXX-0001	DAY 13	EVENING	99	99	99	99	99
XXX-0001	DAY 14	MORNING	2	1	1	2	2
XXX-0001	DAY 14	EVENING	2	2	1	2	3
XXX-0001	DAY 15	EVENING	2	0	1	2	3
XXX-0001	DAY 16	EVENING	1	1	1	3	3
XXX-0001	DAY 17	EVENING	2	1	0	3	3
XXX-0001	DAY 18	EVENING	1	2	1	3	3
XXX-0001	DAY 19	EVENING	1	2	1	3	3
XXX-0001	DAY 20	EVENING	1	2	1	3	3
XXX-0001	DAY 21	EVENING	2	1	2	2	2

Figure 4.4 – Example of imputed result for PHANTOM records

In Figure 4.5, you can see the final appearance of the PHANTOM record in the dataset. It is important to note that the ADT for these records is taken from SDTM.SV, as we don't have the date collected in the diary. In addition, the ATM is imputed as "00:00:00" for MORNING records and "23:59:59" for EVENING ones.

USUBJID	PARAMCD	AVAL	ADTM	ADY	ATPT	AVISIT	AVISITN	DTYPE
XXX-0001	QS00101	2	11MAY2021:09:00:53	10	MORNING	DAY 10	10	
XXX-0001	QS00101	2	11MAY2021:20:01:04	10	EVENING	DAY 10	10	
XXX-0001	QS00101	2	12MAY2021:00:00:00	11	MORNING	DAY 11	11	PHANTOM
XXX-0001	QS00101	2	12MAY2021:21:02:42	11	EVENING	DAY 11	11	

Figure 4.5 – Example of PHANTOM record in ADaM

To ensure that the PHANTOM records were added correctly, we need to check if the number of observations is a multiple of the number of questions in the diary. In our case, there are 14 questions in the diary, so the number of observations should also be 14. This is because we create a PHANTOM record for each of the 14 missing assessments.

MINGLE WITH BASELINE AND DAY 1 DATA

There is a tricky case in the process of creation of Day 1 records. Suppose we defined Baseline visits according to standard baseline rule (the latest assessment before the study treatment start). Some records from Day 1 may have become a Baseline. If we add all planned visits, as it was shown in the previous sub-section, we'll get additional PHANTOM records for Day 1. That would not be correct, because we had the assessments from Day 1 in the data. In Figure 4.6 you can see an example when Day 1 Morning record became Baseline and no additional record was derived for Day 1 Morning. So, it is important to derive Baseline visits after imputing all planned visits.

USUBJID	PARAMCD	AVAL	ADTM	ADY	ATPT	AVISIT	AVISITN	DTYPE
XXX-0001	QS00101	2	02MAY2021:10:03:52	1	MORNING	Baseline	0	
XXX-0001	QS00101	1	02MAY2021:20:13:52	1	EVENING	DAY 1	1	
XXX-0001	QS00101	1	03MAY2021:10:21:16	2	MORNING	DAY 2	2	
XXX-0001	QS00101	1	03MAY2021:20:01:23	2	EVENING	DAY 2	2	

Figure 4.6 – Day 1 Morning record has become a Baseline.

TEMPLATE WITH ALL NEEDED TIME POINTS

To impute all missing responses in the diary data, we need to create a template that includes all necessary analysis visits and time points for each subject and question. Code 2 below contains a macro that can be used to generate the structure of visits and time points. The number of these can be adjusted using the macro parameters.

```
%macro make_templ(days=29, tpts=2, cond_excl=%str(1 ne 1));
   data templ;
```

```
      length avisit $200 atpt $40;
      %do i = 1 %to &days.;
         avisitn=&i.; avisit="Day &i.";
         %do j=1 %to &tpts.;
            atptn=&j.; atpt = strip(put(atptn, atpt_nam.));
            if not (&cond_excl.) then output;
         %end;
      %end;
    run;
%mend make_templ;

*by default, generates morning and evening for day 1 - 29;
%make_templ;

*generates morning and evening for day 1 - 14, only morning for day 15 - 28;
%make_templ(days=28, tpts=2, cond_excl=%str(avisitn>=15 and atptn=2));
```

Code 4.2 – Producing the frame with necessary visits and time points.

Further, we form the list of subjects that participated in diary data gathering and the list of questions (these are referred to as data sets SUBJECTS and PARAMCDS in Code 4.3). The necessary template is obtained as Cartesian product of tables with subjects, parameters, and visits in conjunction with time points. It is important to note that the number of observations in ALL_RECS dataset should be equal to the product of the number of observations in the TEMPL, SUBJECTS, and PARAMCDS datasets.

```
proc sort data=src nodupkey out =subjects(keep=usubjid); by usubjid; run;

proc sort data=src nodupkey out =paramcds(keep=paramcd); by paramcd; run;

proc sql noprint;
   /*table that contains all needed records*/
   create table all_recs as
select * from subjects cross join paramcds cross join templ
order by usubjid, avisitn, atptn, paramcd;
quit;
```

Code 4.3 – Combining intermediate tables in full template.

In an active study, where new data is being added to the database, it is advisable to limit the previously created template to the latest available visit in the subject's SV. This can be achieved by using Code 4.4. If we do not follow this approach, we may end up imputing missing records that are not actually meant to be collected during the current stage of the study.

```
data last_visits(keep=usubjid visitnum rename=(visitnum=last_visn));
   set src.sv(where=(1<=visitnum<=29));
   by usubjid visitnum;
   if last.usubjid;
run;

data templ_current_state(drop=last_visn);
   merge all_recs(in=dat) last_visits;
   by usubjid;
   if avisitn<=last_visn;
run;
```

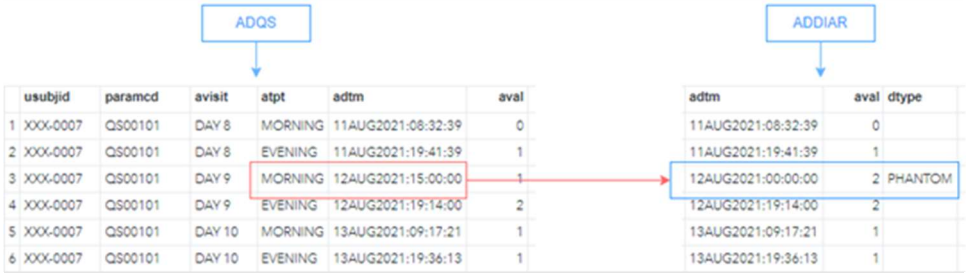
Code 4.4 – Deleting from the template visits that were not performed.

Further, we need to merge the obtained template with the collected data to proceed with AVAL imputation.

4. ADVICE ON ADAM ARRANGEMENTS

It is reasonable to create a separate ADaM (for example, ADDIAR – Diary Analysis Dataset) that will be used as a source for ADTTE. ADDIAR will be based on QS diary data and all missing assessments will be added to it. It can be confusing to impute these records just in ADQS because additional restrictions may apply to the data used in TTE analysis. For instance, suppose we should consider only assessments from 7 a.m. till 12 a.m. as “morning” for TTE analysis. If a subject has a “morning” assessment at 3:00 p.m. (Figure 5.1), it is not valid for TTE analysis and we should derive a “phantom” record for the “morning” time point. Meanwhile, the initial record from 3:00 p.m. should be

kept in ADQS and can be used for other non-TTE outputs to display the results as collected.



	usubjid	paramcd	avisit	atpt	adtm	aval
1	XXX-0007	QS00101	DAY 8	MORNING	11AUG2021:08:32:39	0
2	XXX-0007	QS00101	DAY 8	EVENING	11AUG2021:19:41:39	1
3	XXX-0007	QS00101	DAY 9	MORNING	12AUG2021:15:00:00	1
4	XXX-0007	QS00101	DAY 9	EVENING	12AUG2021:19:14:00	2
5	XXX-0007	QS00101	DAY 10	MORNING	13AUG2021:09:17:21	1
6	XXX-0007	QS00101	DAY 10	EVENING	13AUG2021:19:36:13	1

	adtm	aval	dtype
1	11AUG2021:08:32:39	0	
2	11AUG2021:19:41:39	1	
3	12AUG2021:00:00:00	2	PHANTOM
4	12AUG2021:19:14:00	2	
5	13AUG2021:09:17:21	1	
6	13AUG2021:19:36:13	1	

Figure 5.1 – ADQS vs ADDIAR in case of limitation for time point windows for TTE analysis

Moreover, having intermediate ADaM ADDIAR will facilitate QC process by breaking it up into stages.

5. TIME-TO-EVENT CALCULATION

In this section, we will be discussing the model example that we introduced in the introduction. To summarize, it is about the subjects who fill out a symptom diary (14 questions) 2 times per day in the morning and in the evening for 28 days.

The data related to the symptoms are usually stored in ADQS and captured by the following variables:

- Each symptom corresponds to a single parameter, which is named as PARAMCD.
- The date and time of a symptom assessment is stored in ADTM.
- The visit of an assessment is available in AVISIT.
- Time of assessment is stored in ATPT and can be either Morning or Evening.
- The symptom scores are stored in AVALC, while their numerical counterparts are stored in AVAL.

We are interested in the time of symptom alleviation which is defined as *the time from treatment initiation to the start of the period in which symptom score<=1 and remains <=1 for 21.5 hours.*

To solve this problem, we need to follow three steps. The first step is to calculate the intervals during which the event requires maintenance. In the second step, we need to determine the duration of the maintenance that falls within these intervals and select only those intervals that meet our requirements. Finally, in the third step, we need to select the start time of each qualifying interval.

The first step can be achieved in the following way. Let AVALN indicate whether the symptom is <=1 or not at a single time point where 0 corresponds to the condition being met and 99 means that the condition is not met. Next, we should group such records together if they stand right next to each other and refer to the same subject and symptom. To this end, we introduce the GR variable and assign values to it based on Code 6.1. Whenever we read a new record from the dataset, we add 1 to its previous value. However, if the current record meets the same condition as the previous one and refers to the same subject and symptom, we do not add 1. Thus, two consecutive records have the same value on the condition that both satisfy the condition of the endpoint. Otherwise, two consecutive records always have different values.

```
data adqs1;
   set adqs;
   by usubjid paramcd adtm;

   length avaln prev_avaln prev_adtm 8 prev_usubjid $50 prev_paramcd $8;
   format prev_adtm datetime20.;

   avaln = ifn(aval in (0 1), 0, 99);

   prev_usubjid = lag(usubjid);
   prev_paramcd = lag(paramcd);
   prev_avaln   = lag(avaln);

   gr + 1;

   if  prev_usubjid = usubjid
   and prev_paramcd = paramcd
   and prev_avaln   = avaln
   and avaln = 0
   then gr = gr - 1;
run;
```

Code 6.1 – Identifying the maintenance intervals.

When the intervals of the maintenance of the event are identified, it is high time to select only those where the symptom score is <=1 for 21.5 hours. It can be achieved using the “proc sql” tool presented in Code 6.2.

```
proc sql;
   create table tte1 as
   select *
   from adqs1
   group by usubjid, paramcd, gr
   having (max(adtm) - min(adtm))/3600 >= 21.5
   order by usubjid, paramcd, gr, adtm;
quit;
```

Code 6.2 – Selecting intervals of the needed duration.

Now we have come to the third step of our algorithm, which involves selecting the start of the interval. It can be easily done exploiting `first.` functionality of the data step, see Code 6.3.

```
data tte;
   set tte1;
   by usubjid paramcd gr adtm;
   if first.paramcd;
run;
```

Code 6.3 - Selecting the start of the interval.

CONCLUSION

Dealing with questionnaire data can be challenging. It is essential to maintain data integrity to generate high-quality outputs that can help clinicians with endpoints described in the statistical analysis plan. Following the steps will allow to save time during ADaMs and to keep data quality:

- Defining which diary data will be used for TTE analysis and its preparation in separate dataset.
- Defining strategy for dealing with missing timepoints.
- Establish censoring rules while considering the period of interest.
- Paying attention to defining baseline and its overlap with scheduled assessments.

CONTACT INFORMATION

Ihor Savostianov
Senior Statistical Programming Analyst
Intego Group, LLC
19 Hromadyanska Street
Kharkiv 61057
Ukraine
Work Phone: +38 044 500 7020 ext. 2414
Fax: +38 057 728 3035
Email: igor.savostianov@intego-group.com

Valeriia Oreshko
Senior Statistical Programming Analyst
Intego Group, LLC
35B Borychiv Tik Street
Kyiv 04070
Ukraine
Work Phone: +38 044 500 7020 ext. 2414
Fax: +38 057 728 3035
Email: valeriia.oreshko@intego-group.com