PharmaSUG China 2021 - Paper SR-038

A novel insight into confirmed Best Overall Response per RECIST 1.1

Ji Shuangbin, CR Medicon, Nanjing, China

ABSTRACT

RECIST 1.1 enables researchers a simple and effective way to get solid tumor evaluation. However, calculation of confirmation BOR may be complexed sometimes, especially when there is limitation to number of NE. Here a novel SAS macro provide a simpler way to solve this problem by slicing records into small blocks. Moreover, the established macro is easy to expend and make it possible to deal with some more complexed requirement when needed.

Key words: RECIST 1.1, SAS macro, confirmed BOR, limit to number of NE

INTRODUCTION

Response Evaluation Criteria in Solid Tumors version 1.1 (RECIST 1.1) is the most widely used industry standard of solid tumor in clinical trial currently. Although it is conducted by one-dimension measurement, the procedure of confirmed Best Overall Response (BOR) is still complex sometimes (Bob, 2017). Though a further way is provided to solve the confirmed BOR (Cui, 2020), the procedure seems still complex and counterintuitive. Furth more, macros they provided are not able to figure out problem like certain number of “NE” protocol defined is limited.

Though response values of subjects may be quite different, numerous and complex, however, for the specific subject, the confirmed status should be at least one of COMPLETE RESPONSE (CR), PARTIAL RESPONSE (PR), STABLE DISEASE (SD), PROGRESSIVE DISEASE (PD), and NOT EVALUABLE (NE) occasionally. Since there is a fixed pattern for the order of CR, PR, SD, PD and NE, it is easy to figure out BOR for each subject when all statues have been confirmed before. That is to say, certain subject may be listed with at most four (CR, PR, SD, PD) confirmed statues (those who with no confirmed response will be NE here), but the first for BOR by turns. Therefore, the BOR problem can be simplified as finding “first” response values satisfied the response confirmation definition respectively.

Clinical Data Interchange Standards Consortium (CDISC) provided a series of standards of data linked with healthcare. There, ADRS domain, is designed for efficacy analyses of oncology clinical trials. Here the designed macro ***%ConfBOR*** makes an easy way to get response values from variable AVALC and temporal information, then BOR for each subject will be calculated automatically.

DEFINITION

CR confirmation: A confirmed CR will be listed when duration of two CR records is over confirmation window (defined in protocol, 4 weeks generally) with only CR or NE (limited to certain number as defined in protocol) between the two before first PD.

PR confirmation: A confirmed PR will be listed when duration of the first PR or CR record and the second PR or CR record is over confirmation window with only CR, PR or NE between the two before the first PD.

SD confirmation: A confirmed SD will be listed when at least one record of CR, PR or SD, of which the duration between the record and RFSTDTC (usually the first day of exposure) is not less than the minimized days required (defined in protocol, 6 weeks generally) before the first PD record.

PD confirmation: A confirmed PD will be listed when the first PD record exists.

NE confirmation: A confirmed NE will be listed when subject is not in CR, PR, SD and PD lists.

Actually, in RECIST 1.1, only CR and PR need to be confirmed. Aiming to simplify the model, SD, PD and NE confirmations are specified here.

PROCEDURE

Step 1: Keep all the records before the first PD, including NE (missing value will be treated as NE by default).

Step 2: List all subjects to the CR, PR, SD and PD lists respectively based on the definition mentioned above.

Step 3: Search CR, PR, SD, PD tables to confirm the BOR of certain subject by turns. The very first table show the appearance of subject will be assigned the BOR value of the subject, for those who are not in any lists above will be assigned NE.

Step 4: If interim analysis (IA) is conducted and unconfirmed PR or CR is needed, the BOR from step 3 will be compared with the last effective record (NE is not regarded as an effective record here by default).

Records				No limitation to NE		No more than one NE	
Days 14	Days 42	Days 70	Days 98	BOR	IA	BOR	IA
CR	CR	PD		CR	CR	CR	CR
PR	NE	NE	CR	PR	PR	SD	uCR
PR	NE	PR	PD	PR	PR	PR	PR
PR	CR	NE	NE	PR	uCR	PR	PR

Table 1: Some examples of BOR with or without limitation to NE

For instance, when records of certain subject are “CR-CR-PD” on days 14, 42 and 70, the subject will be listed in CR, SD and PD lists according to the definition above, however, since BOR is confirmed by turns, thus it is still CR for the subject eventually. When the records are “PR-NE-NE-CR” on days 14, 42, 70 and 98, if the limitation to NE is no more than one, the BOR is SD, however, if there is no limitation to NE, BOR should be PR. By the way, for Interim analysis, the limitation is still taking into consideration, last record of “SD-PR-CR-NE-NE” will be NE when limitation is no more than one, but CR when there is no limitation to number of NE.

THE %CONFBOR MACRO PER RECIST 1.1

PARAMETERS

As showed in Figure 1, all parameters will be specified below.

Dsin: The data set contains unique subject ID (USUBJID), response values (variable can be specified by parameter avalc), the evaluate date and the reference start day with structure like ADRS.

List: “LIST” contains the values of response in the order from CR to NE separated by “|”. This paramter should be paired with values in “avalc” and the output result values will be the same with the values in “list”. The value is just two-character short-cut here by default.

Avalc: Usually AVALC will be the “avalc” by default, but if modification is needed before BOR caculated, then assgining other vairable name to “avalc” will be fine. But all the values should be consistent with “list”.

Rfst_dt: A numeric date variable of reference start day for each subject, once left empty, TRTSDT will be used by default.

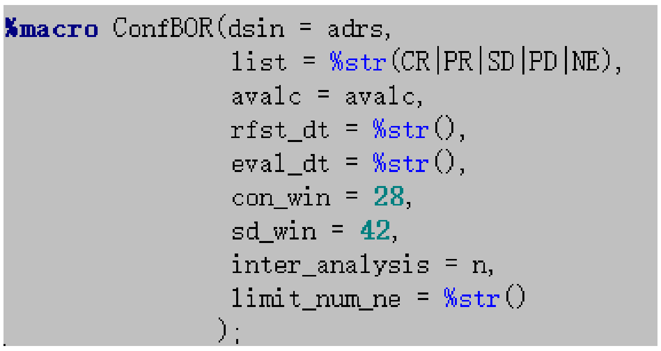
Eval_dt: A numeric evaluation date of response. If not specified, ADT will be regarded as evaluation date of each record.

Con_win: “CON_WIN” is short for “confirmation window”. It is the minimized duration of response confirmation by days defined in protocol, 28 is the default value.

Sd_win: “SD_WIN” is short for “SD window”, which means the minimized days for a “STABLE DISEASE” status, 42 by default.

Inter_analysis: If the value is “y” or “yes”, a variable IA will be output after BOR for interim analysis.

Limit_num_ne: Whether number of NE in data set should be limited for BOR confirmation. “%str()” means there is no limitation, and the specific number means maximized number of NE for BOR confirmation. For example, “limit_num_ne = 0” means there should be no NE between the two records that are remained to confirm.

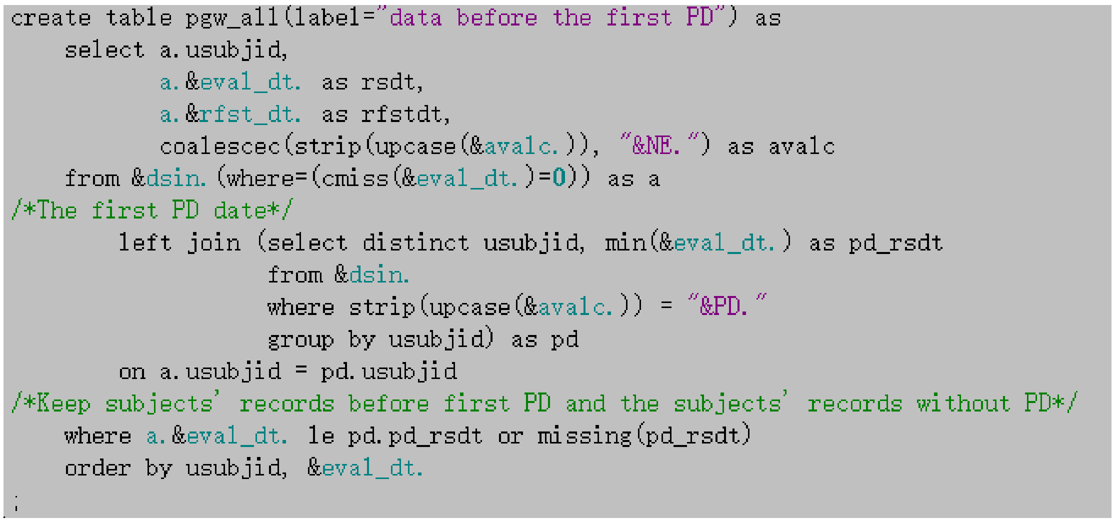


```
%macro ConfBOR(dsin = adrs,
               list = %str(CR|PR|SD|PD|NE),
               avalc = avalc,
               rfst_dt = %str(),
               eval_dt = %str(),
               con_win = 28,
               sd_win = 42,
               inter_analysis = n,
               limit_num_ne = %str()
              );
```

Figure 1. Parameters in *%ConfBOR* SAS macro.

STEP 1

As mentioned above (Figure 2), the very first date of PD is marked for the subject who did have a PD. It is obvious all records before the first PD and the PD records (for PD list below) are kept. For subjects without PD records where the PD_RSDTC is missing, entire observations should be output.

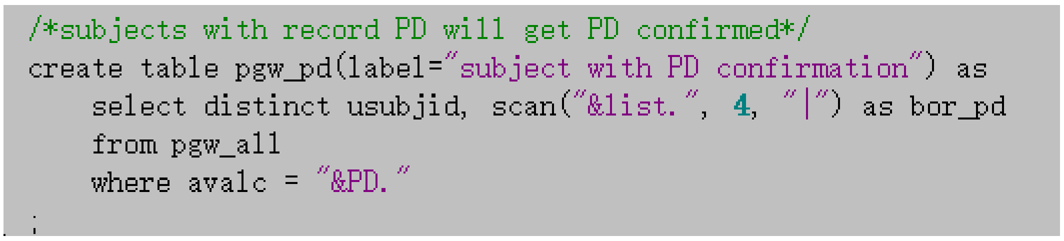


```
create table pgw_all(label="data before the first PD") as
    select a.usubjid,
           a.&eval_dt. as rsdt,
           a.&rfst_dt. as rfstdt,
           coalescec(strip(upcase(&avalc.)), "&NE.") as avalc
    from &dsin.(where=(cmiss(&eval_dt.)=0)) as a
/*The first PD date*/
        left join (select distinct usubjid, min(&eval_dt.) as pd_rsdt
                   from &dsin.
                   where strip(upcase(&avalc.)) = "&PD."
                   group by usubjid) as pd
        on a.usubjid = pd.usubjid
/*Keep subjects' records before first PD and the subjects' records without PD*/
    where a.&eval_dt. le pd.pd_rsdt or missing(pd_rsdt)
    order by usubjid, &eval_dt.
;
```

Figure 2. Records before first PD or records of subjects without PD are kept.

STEP 2

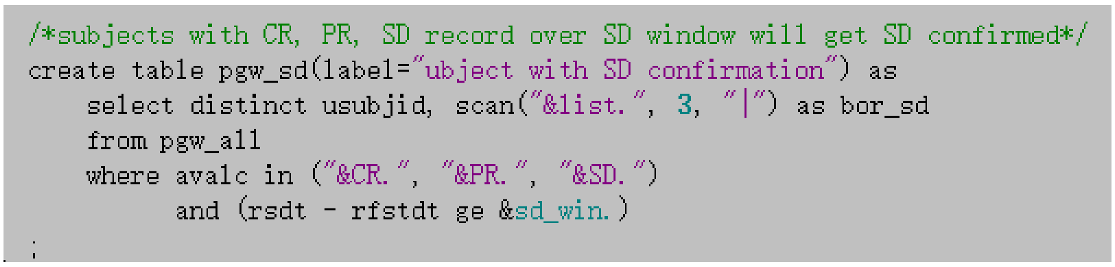
Step 2-1. For PD confirmation (Figure 3).



```
/*subjects with record PD will get PD confirmed*/
create table pgw_pd(label="subject with PD confirmation") as
    select distinct usubjid, scan("&list.", 4, "|") as bor_pd
    from pgw_all
    where avalc = "&PD."
;
```

Figure 3. Output the subjects with PD record from data set PGW_ALL.

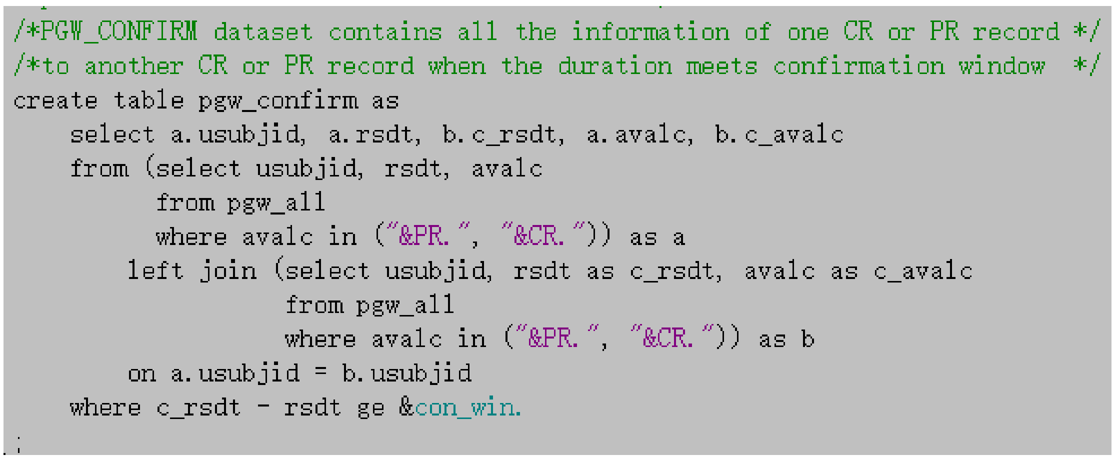
Step 2-2. For SD confirmation (Figure 4).



```
/*subjects with CR, PR, SD record over SD window will get SD confirmed*/
create table pgw_sd(label="ubject with SD confirmation") as
    select distinct usubjid, scan("&list.", 3, "|") as bor_sd
    from pgw_all
    where avalc in ("&CR.", "&PR.", "&SD.")
          and (rsdt - rfstdt ge &sd_win.)
;
```

Figure 4. According to the definition of SD confirmation, CR, PR and SD records with duration over SD window will be output.

Step 2-3. For CR and PR confirmation



```
/*PGW_CONFIRM dataset contains all the information of one CR or PR record */
/*to another CR or PR record when the duration meets confirmation window  */
create table pgw_confirm as
    select a.usubjid, a.rsdt, b.c_rsdt, a.avalc, b.c_avalc
    from (select usubjid, rsdt, avalc
          from pgw_all
          where avalc in ("&PR.", "&CR.")) as a
        left join (select usubjid, rsdt as c_rsdt, avalc as c_avalc
                   from pgw_all
                   where avalc in ("&PR.", "&CR.")) as b
        on a.usubjid = b.usubjid
    where c_rsdt - rsdt ge &con_win.
;
```

Figure 5. Cross link CR and (or) PR records among the same subjects.

First of all, all the CR (PR) record is cross linked with another CR (PR) record of the same subject when the duration is over confirmation window (Figure 5). All eligible records are permutated and combined to retain the information for confirmation. Actually, all records will be divided into small "blocks". For instance, a subject with eligible records "PR-SD-PR-CR" generates "PR-PR", "PR-CR" and a second "PR-CR" three blocks with start and end information (Figure 5, Figure 6).

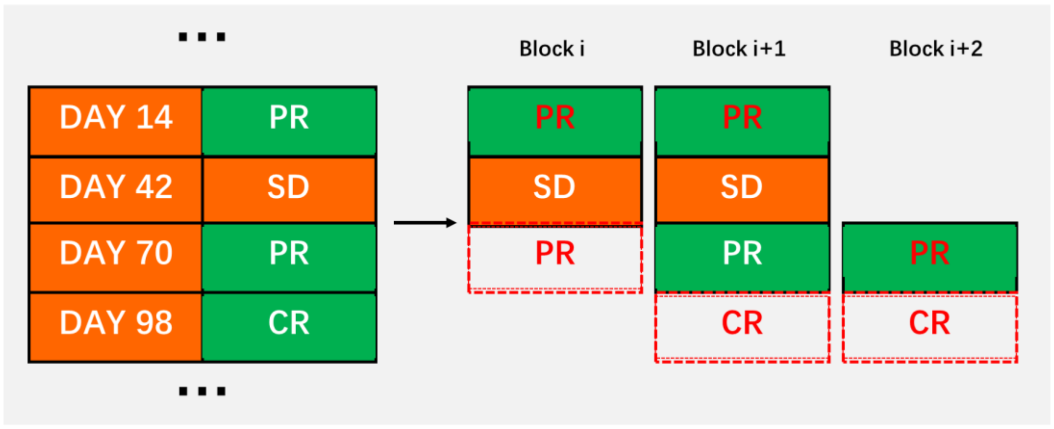


Figure 6. Schematic diagram for block slicing.

Then macro variables are needed to transfer all the information (Figure 7), so that all "blocks" are able to output to data set by using these information (Figure 8).

```
/*Utilize macro variables to store and transfer the information*/
/*of one CR(PR) to another CR(PR)*/
select usubjid, rsdt, c_rsdt, count(*)
into :pr_usubjid1-, :pr_stdt1-, :pr_endt1-, :pr_count trimmed
from pgw_confirm
where not (avalc = "&CR." and c_avalc = "&CR.")
;

select usubjid, rsdt, c_rsdt, count(*)
into :cr_usubjid1-, :cr_stdt1-, :cr_endt1-, :cr_count trimmed
from pgw_confirm
where avalc = "&CR." and c_avalc = "&CR."
;
```

Figure 7. Macro variables for transferring information, where _STDT_ represents date of the "first" CR (PR) and _ENDT_ represents date of the "second" CR (PR).

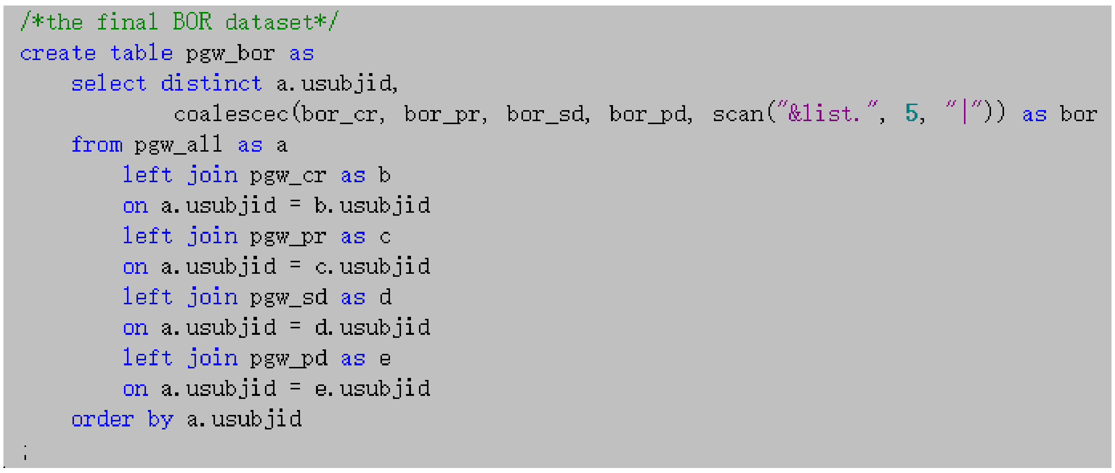
```
data pgw_&pre._out;
    set pgw_all end=last;
    %do i = 1 %to &&&pre._count.;
    if strip(usubjid) = strip("&&&pre._usubjid&i.") and
       &&&pre._stdt&i. le rsdt lt &&&pre._endt&i. then do;
        block = &i.;
        output;
    end;
    %end;
/*in case no block output for CR and PR records*/
    if last then do;
        block = .;
        call missing(of _all_);
        output;
    end;
run;

proc sql;
    create table pgw_&pre.(label="subject with %upcase(&pre.) confirmation") as
        select distinct a.usubjid, "&BOR." as bor_&pre.
        from (select distinct usubjid, block, count(*) as count
              from pgw_&pre._out
              group by usubjid, block) as a
            left join (select distinct usubjid, block, count(*) as count
                       from pgw_&pre._out
                       where &cond.
                       group by usubjid, block) as b
            on a.usubjid = b.usubjid and a.block = b.block
            left join (select distinct usubjid, block, count(*) as count
                       from pgw_&pre._out
                       where avalc = "&NE."
                       group by usubjid, block) as c
            on a.usubjid = c.usubjid and a.block = c.block
        where a.count = b.count
              %if &limit_num_ne. ne %str() %then
                  and c.count le &limit_num_ne.;
    ;
```

Figure 8. Inner macro %GetConfirm for CR and PR confirmation.

Considering procedure of CR and PR confirmation is similar, an inner macro *%GetConfirm* is wrote for CR (PR) confirmation data set (Figure 8). The outline of this macro is utilizing macro variables above to output blocks to data set at the very first. Then CR and PR confirmation is based on these “blocks”. For example, a subject with a confirmed CR must be a subject with at least two CR records, exactly all the macro variables which are output for CR above (Figure 7). Besides only CR or NE (limit to certain number when necessary) record(s) can be in between of the two. Here the first CR records and the records between them are in the same block, therefore when all records count of the block equates to the CR and NE records count, indicating there is a confirmed CR.

STEP 3



```
/*the final BOR dataset*/
create table pgw_bor as
    select distinct a.usubjid,
            coalescec(bor_cr, bor_pr, bor_sd, bor_pd, scan("&list.", 5, "|")) as bor
    from pgw_all as a
        left join pgw_cr as b
        on a.usubjid = b.usubjid
        left join pgw_pr as c
        on a.usubjid = c.usubjid
        left join pgw_sd as d
        on a.usubjid = d.usubjid
        left join pgw_pd as e
        on a.usubjid = e.usubjid
    order by a.usubjid
;
```

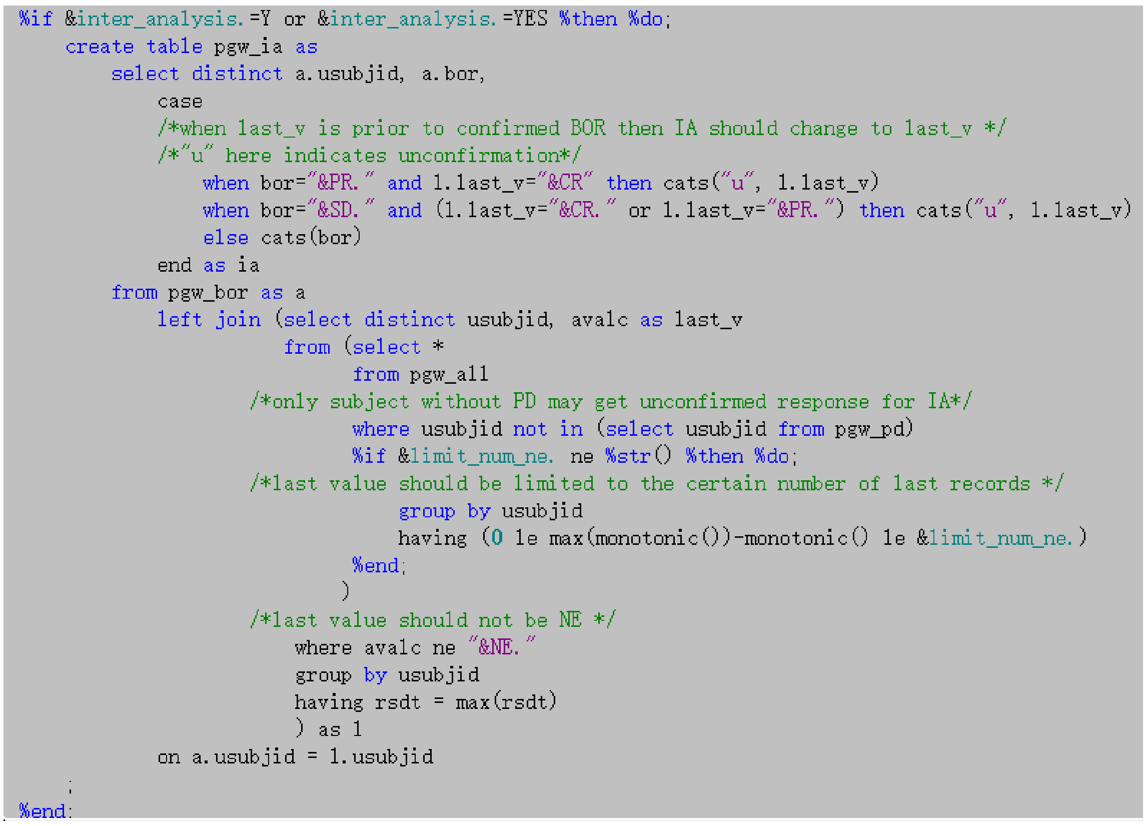
Figure 9. Outputting confirmed BOR based on CR, PR, SD and PD lists by turns.

So far, all statuses have been confirmed individually, and a certain subject may be listed in multiple data sets. However, BOR is confirmed by first appeared status list for all subjects. For those who are not in any confirmed lists will be assigned NE. Here COALESCEC function may help to determine the first confirmed prior status (Figure 9).

Thus, all data sets are put together to output final PGW_BOR data set, where USUBJID is paired with BOR confirmed.

STEP 4

Once interim analysis is conducted, then the output data set includes variable USUBJID, BOR and IA, and the IA may contain “uCR” and (or) “uPR” where “u” is short for “unconfirmed”. When the number of NE is limited, the last effective value should be limited at the same time (Figure 10).



```
%if &inter_analysis.=Y or &inter_analysis.=YES %then %do;
    create table pgw_ia as
        select distinct a.usubjid, a.bor,
            case
            /*when last_v is prior to confirmed BOR then IA should change to last_v */
            /*"u" here indicates unconfirmation*/
                when bor="&PR." and l.last_v="&CR" then cats("u", l.last_v)
                when bor="&SD." and (l.last_v="&CR." or l.last_v="&PR.") then cats("u", l.last_v)
                else cats(bor)
            end as ia
        from pgw_bor as a
            left join (select distinct usubjid, avalc as last_v
                       from (select *
                             from pgw_all
                    /*only subject without PD may get unconfirmed response for IA*/
                             where usubjid not in (select usubjid from pgw_pd)
                             %if &limit_num_ne. ne %str() %then %do;
                    /*last value should be limited to the certain number of last records */
                                 group by usubjid
                                 having (0 le max(monotonic())-monotonic() le &limit_num_ne.)
                             %end;
                            )
                    /*last value should not be NE */
                        where avalc ne "&NE."
                        group by usubjid
                        having rsdt = max(rsdt)
                        ) as l
            on a.usubjid = l.usubjid
    ;
%end;
```

Figure 10. Last effective response is taken into consideration for interim analysis.

DISCUSSION

Macro parameters here can be adjusted when needed. Variable AVALC itself in ADRS data set may contains more results than response values, then a parameter like “PRARM” or “PARAMCD” will make sense to filter original data, or the input “&DSIN” should have been treated previously. Checking every single value in variable AVALC that is just contained in the values of “&LIST” before BOR is calculated is necessary to avoid the situation where pretreatment of AVALC is ignored. Otherwise, an extreme case in which all other values is missing and limitation to NE is performed may lead to an entire wrong result.

Missing value should be taken into consideration. Here in the macro, all “EVAL_DT” with missing value are omitted, and the records are excluded when BOR is calculated. When imputation of evaluation date is actually performed then the new value must be named “ADT” or it should be clarified with parameter EVAL_DT. In general, there should be no missing value in reference start day, thus this macro does not take it into consideration at the moment. If RFST_DT does contain a missing value, a “NOTE” may output in the macro because of the missing value.

Though process of PR and CR confirmation is similar, but treatments of CR and PR confirmation are able to be different. It is feasible to set different limitations to the two. For example, limiting number of NE for PR confirmation individually is reasonable sometimes. Any PR or SD record after CR should be regarded as data issue, and data check should be performed for CR individually. When macro *%GetConfirm* is executed, count of all records and count of records with only CR and NE is checked grouped by blocks. In fact, all subjects with two equal counts are kept. However, those blocks with the two are different are indeed the data with issue, and a PGW_CHECK data set just remains this result. So far, all CR records checks before first PD are carried out. If entire CR response, whether or not before first PD, requires check, PGW_CHECK would be derived from original input data set where the first CR should be taken into special consideration.

In actual application process, NOT AVALIABLE (NA) is usually contained in the CRF for some special cases in which response values could not be clarified. Then the definition of CR and PR confirmation might be adjusted as description in protocol. Correspondingly, code of CR and PR confirmation should take “NA” into consideration. Even though, core part of codes is still to slice original records into small blocks just like the way NE are treated.

CONCLUSION

This developed macro is able to calculate BOR according to RECIST 1.1 with a relatively simple and intuitive way. Meanwhile, it is recommended a ADRS data set as input or a data set with the same structure of ADRS. Moreover, this macro makes it possible to calculate BOR with a limitation to number of NE. Since all four responses are confirmed separately, any limitation is able to be set for calculating BOR when necessary. All in all, this paper provides a novel insight into confirmed Best Overall Response per RECIST 1.1.

REFERENCE

Eisenhauera, E.A., Therasseb, P., Bogaertsc, J., Schwartzd, L.H., Sargente, D., Fordf, R., Danceyg,J., Arbuckh, S., Gwytheri, S., Mooneyg, M., Rubinsteing, L., Shankarg, L., Doddg, L., Kaplanj, R.,Lacombec, D., Verweijk, J. 2009. “New response evaluation criteria in solid tumours: Revised RECIST guideline (version 1.1).” *European Journal of Cancer*, 45:228 – 247.

Bob Zhong, Jiangfan Li, Hong Xie, Peter De Porre, Kenneth Maahs, Kyounghwa Bae. 2017. “SAS Macro for Derivation of Best Overall Response per RECIST 1.1.” *PharmaSUG 2017.* Available at https://www.pharmasug.org/proceedings/2017/AD/PharmaSUG-2017-AD24.pdf

Xiangchen (Bob) Cui, and Sri Pavan Vemuri. 2020. “Simplifying the Derivation of Best Overall Response per RECIST 1.1 and iRECIST in Solid Tumor Clinical Studies.” *PharmaSUG 2020*. Available at https://www.lexjansen.com/pharmasug/2020/DV/PharmaSUG-2020-DV-066.pdf

ACKNOWLEDGMENTS

Author would like to thank Qi Meng and Kejin Liu for their suggestions on this manuscript.

CONTACT INFORMATION

Name: Ji Shuangbin
E-mail: shuangbin.ji@crmedicon.com

Appendix

```
/*************************************************************************************/
/*                                        Description                                */
/* This macro is designed for confirmed BOR calculation based on RECIST 1.1              */
/* The structure like CDISC datasets ADRS with USUBJID, reference date, evaluation date are needed */
/* Response value should be assigned CR|PR|SD|PD|NE or alternative values which are the same in dataset */
/* Confirmation window for the adjacent response should be specified according to protocol          */
/* A special window for STABLE DISEASE (SD) confirmation is needed                        */
/* It is alternative for interim analysis where BOR may assigned by uCR or uPR               */
/* Limitation to number of NOT EVALUABLE (NE) between two response records are able to clarify      */

/*************************************************************************************/

%macro ConfBOR(dsin = adrs,
                   list = %str(CR|PR|SD|PD|NE),
                   avalc = avalc,
                   rfst_dt = %str(),
                   eval_dt = %str(),
                   con_win = 28,
                   sd_win = 42,
                   inter_analysis = n,
                   limit_num_ne = %str()
                  );

/*************************************************************************************/
/*                               Paramaters and input check                            */
/*************************************************************************************/

    %local CR PR SD PD NE avalclist tempaval dsid rc check_nobs;

    %let dsin = %sysfunc(dequote(%unquote(&dsin.)));
    %let list = %sysfunc(dequote(%unquote(&list.)));
    %let avalc = %sysfunc(dequote(%unquote(&avalc.)));
    %let con_win = %sysfunc(dequote(%unquote(&con_win.)));
    %let sd_win = %sysfunc(dequote(%unquote(&sd_win.)));
    %let inter_analysis = %upcase(%sysfunc(dequote(%unquote(&inter_analysis.))));
    %let limit_num_ne = %sysfunc(compress(%sysfunc(dequote(&limit_num_ne.)),    , KD));

    %if %length(&dsin.) = 0 or %length(&list.) = 0 or %length(&avalc.) = 0
        or %length(&con_win.) = 0 or %length(&sd_win.) = 0 %then %do;
        %put Err%str()or: Paramters dsin, list, avalc, con_win and sd_win can not be null;
        %goto exit;
    %end;

    %if %sysfunc(exist(&dsin.))=0 %then %do;
        %put Err%str()or: dataset &dsin. and &adsl. must exist;
        %goto exit;
    %end;

    %let dsid = %sysfunc(open(&dsin., i));

    %if %sysfunc(varnum(&dsid., usubjid))=0 or %sysfunc(varnum(&dsid., &avalc.))=0 %then %do;
        %put Err%str()or: variable usubjid, &avalc. must exist in &dsin.;
        %goto exit;
    %end;

    %if &eval_dt. = %str() %then %do;
        %put Ifo: Parameter EVAL_DT is null, ADT will be regarded as evaluation date.;
        %let eval_dt = adt;
    %end;
    %else %let eval_dt = %sysfunc(dequote(%unquote(&eval_dt.)));
    %if %sysfunc(varnum(&dsid., &eval_dt.))=0 %then %do;
        %put Err%str()or: evaluation date (&eval_dt.) must exist in &dsin.;
        %goto exit;
    %end;

    %if &rfst_dt. = %str() %then %do;
        %put Ifo: Parameter RFST_DT is null, TRTSDT will be regarded as reference start day.;
        %let rfst_dt = trtsdt;
    %end;
    %else %let rfst_dt = %sysfunc(dequote(%unquote(&rfst_dt.)));
    %if %sysfunc(varnum(&dsid., &rfst_dt.))=0 %then %do;
        %put Err%str()or: reference date (&rfst_dt.) must exist in &dsin.;
        %goto exit;
    %end;

    %let rc=%sysfunc(close(&dsid.));

    %if %sysfunc(countw(&list., %str(|))) ne 5 %then %do;
        %put Err%str()or: Paramter list should show values of CR, PR, SD, PD and NE in &avalc. by turns;
        %goto exit;
    %end;
    %let CR = %upcase(%sysfunc(strip(%scan(&list., 1, %str(|)))));
    %let PR = %upcase(%sysfunc(strip(%scan(&list., 2, %str(|)))));
    %let SD = %upcase(%sysfunc(strip(%scan(&list., 3, %str(|)))));
    %let PD = %upcase(%sysfunc(strip(%scan(&list., 4, %str(|)))));
    %let NE = %upcase(%sysfunc(strip(%scan(&list., 5, %str(|)))));

    proc sql noprint;
        select distinct strip(upcase(&avalc.)) into :avalclist separated by "|"
        from &dsin.
        where not missing(&avalc.)
        ;

    %do i = 1 %to %sysfunc(countw(&avalclist., |));
        %let tempaval = %upcase(%scan(&avalclist., &i., |));
        %if not ("&tempaval." = "&CR." or "&tempaval." = "&PR." or "&tempaval." = "&SD."
                or "&tempaval." = "&PD." or "&tempaval." = "&NE.") %then %do;
            %put War%str()ning: Values "&tempaval." in &avalc. and list "&list." must be consistent;
            %goto exit;
        %end;
    %end;

/*******************************************************************************************/
/*                              Step 1: Output all records before first PD                   */
/*******************************************************************************************/
        create table pgw_all(label="data before the first PD") as
            select a.usubjid,
                   a.&eval_dt. as rsdt,
                   a.&rfst_dt. as rfstdt,
                   coalescec(strip(upcase(&avalc.)), "&NE.") as avalc
            from &dsin.(where=(cmiss(&eval_dt.)=0)) as a
        /*The first PD date*/
                left join (select distinct usubjid, min(&eval_dt.) as pd_rsdt
                           from &dsin.
                           where strip(upcase(&avalc.)) = "&PD."
                           group by usubjid) as pd
                on a.usubjid = pd.usubjid
        /*Keep subjects' records before first PD and the subjects' records without PD*/
            where a.&eval_dt. le pd.pd_rsdt or missing(pd_rsdt)
            order by usubjid, &eval_dt.
        ;
```

```
/*******************************************************************************************/
/*                              Step 2: Output    CR, PR, SD and PD lists respectively                              */
/*******************************************************************************************/

/*2-1 Out put PD and SD>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*/
        /*subjects with record PD will get PD confirmed*/
        create table pgw_pd(label="subject with PD confirmation") as
            select distinct usubjid, scan("&list.", 4, "|") as bor_pd
            from pgw_all
            where avalc = "&PD."
        ;

        /*subjects with CR, PR, SD record over SD window will get SD confirmed*/
        create table pgw_sd(label="ubject with SD confirmation") as
            select distinct usubjid, scan("&list.", 3, "|") as bor_sd
            from pgw_all
            where avalc in ("&CR.", "&PR.", "&SD.") and (rsdt - rfstdt ge &sd_win.)
        ;

/*2-2 Out put CR and PR>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*/
        /*PGW_CONFIRM dataset contains all the information of one CR or PR record */
        /*to another CR or PR record when the duration meets confirmation window    */
        create table pgw_confirm as
            select a.usubjid, a.rsdt, b.c_rsdt, a.avalc, b.c_avalc
            from (select usubjid, rsdt, avalc
                  from pgw_all
                  where avalc in ("&PR.", "&CR.")) as a
                left join (select usubjid, rsdt as c_rsdt, avalc as c_avalc
                           from pgw_all
                           where avalc in ("&PR.", "&CR.")) as b
                on a.usubjid = b.usubjid
            where c_rsdt - rsdt ge &con_win.
        ;

        /*Utilize macro variables to store and transfer the information*/
        /*of one CR(PR) to another CR(PR)*/
        select usubjid, rsdt, c_rsdt, count(*)
            into :pr_usubjid1-, :pr_stdt1-, :pr_endt1-, :pr_count trimmed
            from pgw_confirm
            where not (avalc = "&CR." and c_avalc = "&CR.")
        ;

        select usubjid, rsdt, c_rsdt, count(*)
            into :cr_usubjid1-, :cr_stdt1-, :cr_endt1-, :cr_count trimmed
            from pgw_confirm
            where avalc = "&CR." and c_avalc = "&CR."
        ;
    quit;

        /*Macro GetConfirm output confirmation blocks first, then the CR and PR lists*/
    %macro GetConfirm(pre=pr, cond=%str(avalc in ("&CR.", "&PR.", "&NE.")));
        %local BOR;
        %if &pre. = pr %then
            %let BOR = %scan(&list., 2, %str(|));
        %else %let BOR = %scan(&list., 1, %str(|));

        data pgw_&pre._out;
            set pgw_all end=last;
            %do i = 1 %to &&&pre._count.;
            if strip(usubjid) = strip("&&&pre._usubjid&i.") and
               &&&pre._stdt&i. le rsdt lt &&&pre._endt&i. then do;
                block = &i.;
                output;
            end;
            %end;
        /*in case no block output for CR and PR records*/
            if last then do;
                block = .;
                call missing(of _all_);
                output;
            end;
        run;

        proc sql;
            create table pgw_&pre.(label="subject with %upcase(&pre.) confirmation") as
                select distinct a.usubjid, "&BOR." as bor_&pre.
                from (select distinct usubjid, block, count(*) as count
                      from pgw_&pre._out
                      group by usubjid, block) as a
                    left join (select distinct usubjid, block, count(*) as count
                               from pgw_&pre._out
                               where &cond.
                               group by usubjid, block) as b
                    on a.usubjid = b.usubjid and a.block = b.block
                    left join (select distinct usubjid, block, count(*) as count
                               from pgw_&pre._out
                               where avalc = "&NE."
                               group by usubjid, block) as c
                    on a.usubjid = c.usubjid and a.block = c.block
                where a.count = b.count
                      %if &limit_num_ne. ne %str() %then
                          and c.count le &limit_num_ne.;
            ;
        quit;
    %mend GetConfirm;

    %GetConfirm(pre=pr, cond=%str(avalc in ("&CR.", "&PR.", "&NE.")));
    %GetConfirm(pre=cr, cond=%str(avalc in ("&CR.", "&NE.")));

/*****************************************************************************************/
/*                    Step 3: Search CR, PR, SD, PD lists by turns for BOR               */
/*****************************************************************************************/
    proc sql;
        /*the final BOR dataset*/
        create table pgw_bor as
            select distinct a.usubjid,
                    coalescec(bor_cr, bor_pr, bor_sd, bor_pd, scan("&list.", 5, "|")) as bor
            from pgw_all as a
                left join pgw_cr as b
                on a.usubjid = b.usubjid
                left join pgw_pr as c
                on a.usubjid = c.usubjid
                left join pgw_sd as d
                on a.usubjid = d.usubjid
                left join pgw_pd as e
                on a.usubjid = e.usubjid
            order by a.usubjid
        ;

/***************************************************************************************/
/*                    Step 4: For interim analysis when needed                          */
/***************************************************************************************/
    %if &inter_analysis.=Y or &inter_analysis.=YES %then %do;
        create table pgw_ia as
```

```
            select distinct a.usubjid, a.bor,
                case
                /*when last_v is prior to confirmed BOR then IA should change to last_v */
                /*"u" here indicates unconfirmation*/
                    when bor="&PR." and l.last_v="&CR" then cats("u", l.last_v)
                    when bor="&SD." and (l.last_v="&CR." or l.last_v="&PR.") then cats("u", l.last_v)
                    else cats(bor)
                end as ia
            from pgw_bor as a
                left join (select distinct usubjid, avalc as last_v
                           from (select *
                                 from pgw_all
                        /*only subject without PD may get unconfirmed response for IA*/
                                 where usubjid not in (select usubjid from pgw_pd)
                                 %if &limit_num_ne. ne %str() %then %do;
                        /*last value should be limited to the certain number of last records */
                                     group by usubjid
                                     having (0 le max(monotonic())-monotonic() le &limit_num_ne.)
                                 %end;
                                )
                        /*last value should not be NE */
                            where avalc ne "&NE."
                            group by usubjid
                            having rsdt = max(rsdt)
                            ) as l
                on a.usubjid = l.usubjid
        ;
    %end;
    quit;

/*************************************************************************************/
/*                          Records check for records after CR                      */
/*************************************************************************************/

        /*output the check blocks*/
    data pgw_all;
        set pgw_all;
        by usubjid;
        retain check_block 0;
        if first.usubjid then
            check_block = 0;
        if avalc = "&CR." then
            check_block = check_block + 1;
    run;

    proc sql;
        /*if non-CR, non-NE records after a CR, that may be a data issue*/
        create table pgw_check(label="CR value check") as
            select a.*
            from (select *, count(*) as count
                  from pgw_all
        /*keep records after first CR record in PGW_ALL dataset*/
                  where check_block gt 0
                  group by usubjid, check_block) as a
                left join (select distinct usubjid, check_block, count(*) as count
                           from pgw_all
                           where avalc in ("&CR.", "&NE.")
                           group by usubjid, check_block) as b
                on a.usubjid = b.usubjid and a.check_block = b.check_block
        /*when count of all dose not equal to count of CR, NE, output*/
            where a.count ne b.count
        ;
```

```
     quit;

     %let dsid = %sysfunc(open(pgw_check, in));
     %let check_nobs = %sysfunc(attrn(&dsid., nobs));
     %let rc = %sysfunc(close(&dsid.));
     %if &check_nobs. ne 0 %then
          %put War%str()ning: Check the results of data check for records after CR;

     %exit:

%mend ConfBOR;
```