PharmaSUG 2016 - Paper SP06

A Dose Escalation Method for Dual-Agent in Phase 1 Cancer Clinical Trial using the SAS MCMC Procedure

Gwénaël Le Teuff, Gustave Roussy, Paris, France
Mohamed Amine Bayar, Gustave Roussy, Paris, France

ABSTRACT

Continual reassessment method (CRM) is a model-based dose escalation method commonly used to design a phase 1 trial in oncology when evaluating one agent. The main characteristics include the definition of a working model for dose levels, an acceptable level of toxicity (target) and a model defining the dose-toxicity relationship (for example, power or logistic function). This relationship is updated after the toxicity evaluation of each patient in the cohort and we assign to the next cohort the dose level closest to the target. This allows estimating the maximum tolerated dose (MTD). With the advance of targeted therapies era in oncology, more and more phase 1 trials aim to identify one or more MTDs from a set of available dose levels of two or more agents. Combining several agents can indeed increase the overall anti-tumor action but at the same time increase toxicity. Since the single-agent dose finding methods (algorithm-based or model-based) are not appropriate for combination therapies, several authors developed different methods. In this paper, we propose to illustrate the SAS MCMC procedure through 2 examples related to the phase 1 cancer clinical trials. The first example shows how to estimate the model parameters of Bayesian CRM. The second example presents a SAS program we developed to implement a dose escalade method for dual agent.

INTRODUCTION

Continual reassessment method (CRM) developed by O'Quigley et al (O'Quigley) is a Bayesian model-based dose escalation method commonly used for the design of a phase 1 trial in oncology when evaluating one agent. After assuming a working model, an acceptable level of toxicity (target) and a link function between the response and the dose levels of the agent, we estimate the probability of response for each dose level after the assessment of the response for the first cohort of patients. The response is the occurrence (yes/no) of a dose limiting toxicity (DLT) as defined by the protocol. The relationship between dose levels and the probability of a DLT is then updated. We assign patients of the next cohort to the dose level that is closest to the target and so forth. For safety reasons, a no skipping rule is usually established in the dose escalade design. In practice, statisticians involved in Phase 1 trial use the well-known dfcrm R package developed by Cheung (Cheung), including SAS users. When designing a dose escalation trial combining 2 or more agents (which is more and more frequent with the targeted therapies in oncology), several authors proposed more appropriate methods to those used for dose escalation design for one agent (Hirakawa, Riviere). Among them, Wang and Ivanova (Wang) proposed a Bayesian model-based method with 3 parameters. In the first section of this paper, we show how to use SAS MCMC procedure for Bayesian CRM. In the second section, we propose a SAS macro implementing Wang and Ivanova method for dose escalation design combining 2 agents.

PARAMETER ESTIMATION OF CONTINUAL REASSESSMENT METHOD

In this section, we use MCMC procedure (SAS version 9.4) to estimate the model parameters of Bayesian CRM. The aim, here, is not to provide a SAS code implementing the dose escalade design, since it has been already proposed by some authors (Ishizuka) without the need to use MCMC procedure.

Assuming a CRM design with a power function as used by default in dfcrm R package (1).

$$\Pr\{\text{toxicity at } s_i\} = \varphi(s_i, \alpha) = a_i^{\alpha} \quad (1)$$

With a_i $(i: 1, \ldots, 4)$ defining the working model, model parameter, $\beta=\ln(\alpha)$, following a normal distribution $N(0, \sigma^2)$ (σ^2 fixed to 1.34 in dfcrm R package).

Supposing a fictitious data from a phase 1 clinical trial evaluating four dose levels $d \text{ in } (1, 2, 3, 4)$ and including 40 patients. A working model (0.07, 0.17, 0.30, 0.44) and a 30% pre-specified targeted level of toxicity were set. The DLT y is coded 0 or 1.

The following SAS program allows estimating the probabilities of toxicity for each dose level by 2 different approaches. *Approach 1* consists in obtaining the posterior mean of the parameter β and then calculating the probability of toxicity at each dose level by plugging the posterior mean parameter in equation (1). *Approach 2* consists in calculating directly the posterior means of the probability of toxicity at each dose level.

```
data tox_individual;
    d=1;do i=1 to 4;y=0;id=i;output;end;
       d=2;do i=1 to 6;y=0;id+1;output;end;do i=1 to 2;y=1;id+1;output;end;
       d=3;do i=1 to 6;y=0;id+1;output;end;do i=1 to 4;y=1;id+1;output;end;
       d=4;do i=1 to 6;y=0;id+1;output;end;do i=1 to 8;y=1;id+1;output;end;
       label d='dose level' y='dlt (yes/no)';
       drop i;
run;

%macro CRM_MCMC(alpha=0.10,direct_post_prob=);

ods output PostSumInt=PostSumInt;
proc mcmc data=tox_individual stats(alpha=&alpha) seed=779 nbi=1000 nmc=50000
outpost=out plots=none
   %if &direct_post_prob=1 %then %do; monitor=(beta post1 post2 post3 post4) %end;;

    parm beta 0;
    array a[4] (0.07, 0.17, 0.30, 0.44);
    begincnst;
             v=1.34;/* in dfcrm R package scale = sqrt(1.34) */
    endcnst;
    %if &direct_post_prob=1 %then %do;/* Direct computation of posterior probability*/
    beginprior;
       array post(4);
       %do i=1 %to 4;
              post[&i]=a[&i]**exp(beta);
       %end;
    endprior;
    %end;

    prior beta ~ normal(0,var=v);
    llike=(log(a[d]**exp(beta))**y)*(log(1-a[d]**exp(beta))**(1-y));
    model general(llike);
run;
ods output close;

data result&direct_post_prob;
    set PostSumInt;
    if parameter="beta" then do;
       array pre_p[4] _temporary_ (0.07, 0.17, 0.30, 0.44);
       do i=1 to 4;
          dose_level=i;
          post_p=pre_p[i]**exp(Mean);
          post_p_inf=pre_p[i]**exp(HPDUpper);post_p_inf2=pre_p[i]**exp(Mean+probit(1-
(&alpha)/2)*StdDev);
          post_p_sup=pre_p[i]**exp(HPDLower);post_p_sup2=pre_p[i]**exp(Mean-probit(1-
(&alpha)/2)*StdDev);
          output;
       end;
    end;
    else output;
    drop i;
run;

%mend CRM_MCMC;
%CRM_MCMC(direct_post_prob=0);/* approach 1*/
%CRM_MCMC(direct_post_prob=1);/* approach 2*/
```

Table 1 reports the estimation of the toxicity posterior probability using the 2 SAS-based approaches with similar results.

	SAS proc MCMC		dfcrm R package
Dose level	Approach 1	Approach 2	dfcrm
1	0.1252	0.1321	0.125
2	0.2505	0.2538	0.250
3	0.3904	0.3899	0.390
4	0.5266	0.5236	0.526

Table 1. Toxicity posterior probability for power model with the SAS MCMC procedure and using dfcrm *R* package

We compare these results with those obtained with the dfcrm R package running from a SAS 9.4 session. The first step consists in verifying whether SAS can execute R language statement. This is done by running

```
proc options option=rlang;run;
```

If no connexion is found, we have to add the option -rlang in the file allowing to execute SAS as follows:

"C:\Program Files\SASHome\SASFoundation\9.4\sas.exe" -CONFIG "C:\Program Files\SASHome\SASFoundation\9.4\nls\en\sasv9.cfg" -rlang

The second step is to set the environment variable R_HOME to specify the path. With SAS 9.4 the statement is:

```
option set=R_HOME='C:\Program Files\R\R-3.1.2';
```

Another possibility is to add the statement -set R_HOME 'C:\Program Files\R\R-3.1.2' in the sasv9.cfg. The following SAS code calls the dfcrm *R* package from a SAS session.

```
proc iml;

   run ExportDataSetToR("tox_individual","tox_individual");
   submit / R;

   require(dfcrm)
   library(dfcrm)
   target  <- 0.30
   prior <-c(0.07, 0.17, 0.30, 0.44)

   crm_est <-function(tox,level) {
       n     <- length(level)
       res   <- crm(prior, target, tox, level, include=1:n,
method="bayes",model="empiric",conf.level=0.90)
       return(res)
   }

   print(tox_individual)
   names(tox_individual)
   level <- tox_individual$d
   tox   <- tox_individual$y
   crm_est(tox,level)

   endsubmit;

quit;
```

Results obtained using dfcrm R package are similar to those given by the SAS MCMC procedure (Table 1). The posterior mean of β is -0.2469 using SAS MCMC procedure and -0.246 using dfcrm R package. The posterior variance of β is $(0.2257)^2=0.0509$ using SAS and 0.05 using dfcrm R package. The 90% credibility intervals of toxicity probabilities are given in Table 2 with similar results between SAS approaches and dfcrm R package. The previous SAS code can be easily extended when considering a CRM with a logistic link function.

	SAS proc MCMC			dfcrm R package[†]
	Approach 1		Approach 2[‡]	
	Method 1[†]	Method 2[††]		
Dose level				
1	[0.0492;0.2385]	[0.0478; 0.2345]	[0.0382; 0.2204]	[0.049; 0.238]
2	[0.1344; 0.3848]	[0.1319; 0.3805]	[0.1268; 0.3748]	[0.135;0.384]
3	[0.2557; 0.5226]	[0.2524; 0.5186]	[0.2525; 0.5187]	[0.256;0.522]
4	[0.3946; 0.6424]	[0.3912; 0.6391]	[0.3995; 0.6459]	[0.395;0.642]

Table 2. Comparison of the 90 percent credibility intervals for power model using SAS MCMC procedure and dfcrm *R* package

[†]: credibility intervals are obtained from the lower and upper limits of the parameter β calculated as β +/- the 90th percentile of a standard normal distribution x $\sqrt{\text{posterior variance}}$) and exponential transformation

[††]: credibility intervals are obtained from the estimated credibility interval (highest posterior density) of the parameter β and exponential transformation

[‡]: this approach consists in estimating directly the posterior means and credibility intervals (highest posterior density)

WANG AND IVANOVA METHOD FOR DOSE ESCALADE FOR DUAL AGENTS

Wang and Ivanova (Wang) proposed a Bayesian dose escalade method for dual agent where a MTD of one agent s (with dose levels $s_i, i = 1, \dots, I$) was estimated for each dose level $(t_j, j = 1, \dots, J)$ of the second agent t ($I \geq J$). They used a 3-parameter model (2).

$$\varphi(s_i, t_j, \theta) = 1 - (1 - a_i)^{\alpha}(1 - b_j)^{\beta + \gamma \log(1 - a_i)} \qquad (2)$$

With a_i $(i: 1, \dots, I)$ and b_j $(j: 1, \dots, J)$ are constant defining the working model for agent s and t, respectively and model parameter $\theta = (\alpha, \beta, \gamma)$.

The dose escalade method may include or not a start-up. The start-up allows acquiring data for the beginning of the dose escalation (See paper for more details). The authors firstly evaluated the operating characteristics (correct combination selection, patient allocation, and mean number of observed toxicities) of the 2-parameter model i.e. with no interaction ($\gamma = 0$). The parameters (α, β) follow independent exponential distribution with mean 1. We developed a SAS macro implementing this method. To validate our SAS macro, we compared the results obtained to those published in Wang et al. using the same scenarios for the simulation study. Table 3 represents the different scenarios with columns and rows representing the dose levels of first agent s and second agent t, respectively.

	s1	s2	s3	s4	s5	s6
			Scenario 1			
Level 3	0.08	0.13	**0.20**	0.29	0.40	0.53
Level 2	0.05	0.08	0.13	**0.20**	0.29	0.40
Level 1	0.03	0.05	0.08	0.13	**0.20**	0.29
			Scenario 2			
Level 3	0.05	0.08	0.11	0.15	**0.21**	0.29
Level 2	0.04	0.06	0.09	0.13	**0.18**	0.25
Level 1	0.04	0.05	0.08	0.11	0.15	**0.21**
			Scenario 3			
Level 3	**0.20**	0.30	0.41	0.53	0.65	0.70
Level 2	0.10	**0.20**	0.25	0.32	0.41	0.50
Level 1	0.03	0.05	0.13	**0.20**	0.27	0.35
			Scenario 4			
Level 3	**0.20**	0.40	0.47	0.56	0.65	0.76
Level 2	0.08	0.13	**0.20**	0.32	0.41	0.50
Level 1	0.03	0.05	0.08	0.13	0.17	**0.20**
			Scenario 5			
Level 4	**0.20**	0.29	0.40	0.53		
Level 3	0.13	**0.20**	0.29	0.40		
Level 2	0.08	0.13	**0.20**	0.29		
Level 1	0.05	0.08	0.13	**0.20**		

Table 3. Scenarios for a two-agent trial with target probability of toxicity 0.2. Maximum-tolerated combinations (or combinations closest to them) are in bold

Table 4 displays similar results in terms of the percentage of experimentation at each combination.

	Wang's results						SAS macro					
	s1	**s2**	**s3**	**s4**	**s5**	**s6**	**s1**	**s2**	**s3**	**s4**	**s5**	**s6**
							Scenario 1					
Level 3	2	6	**9**	8	2	0	2.11	5.59	**9.33**	8.14	1.88	0.53
Level 2	1	4	11	**14**	6	2	0.79	3.97	10.97	**14.01**	5.91	1.52
Level 1	4	5	8	10	**7**	2	3.73	4.93	7.26	9.99	**6.89**	2.38
							Scenario 2					
Level 3	1	3	6	11	**8**	5	1.35	3.19	5.85	10.67	**7.76**	5.17
Level 2	1	2	5	9	**11**	5	0.73	3.10	5.39	8.71	**10.17**	5.61
Level 1	4	4	5	7	8	**5**	3.73	4.79	5.03	6.11	7.47	**5.14**
							Scenario 3					
Level 3	**5**	6	3	1	0	0	**5.19**	6.44	2.51	0.97	0.17	0.02
Level 2	5	**17**	14	7	2	0	4.43	**18.20**	14.35	7.09	1.50	0.35
Level 1	4	8	14	**9**	3	1	3.78	8.01	14.08	**8.83**	3.07	0.99
							Scenario 4					
Level 3	**4**	5	2	1	0	0	**4.55**	5.32	2.51	1.01	0.19	0.02
Level 2	3	12	**16**	11	4	1	1.84	11.29	**16.72**	12.42	3.54	0.85
Level 1	4	6	11	10	6	**2**	3.75	5.82	10.80	11.14	6.09	**2.12**
							Scenario 5					
Level 4	**5**	5	2	1			**3.39**	4.45	1.09	0.32		
Level 3	6	**10**	5	2			6.09	**10.19**	5.54	1.62		
Level 2	3	11	**15**	8			2.18	13.13	**14.24**	8.18		
Level 1	4	6	11	**8**			3.44	6.46	10.99	**8.68**		

Table 4. Percentages of experimentation. Percentages at the maximum-tolerated combinations (or doses closest to them) are in bold

Table 5 displays similar results in terms of percentages of the final maximum-tolerated combination recommendation i.e. the percentage of dose recommended of the first agent s for each dose level of the second agent t (row percentage). The SAS macro also reports the average number of toxicities (data not shown).

	Wang's results						SAS macro					
	s1	**s2**	**s3**	**s4**	**s5**	**s6**	**s1**	**s2**	**s3**	**s4**	**s5**	**s6**
							Scenario 1					
Level 3	25	23	**33**	18	2	0	20.03	19.95	**35.70**	21.00	3.10	0.23
Level 2	0	4	26	**54**	14	1	0.10	2.55	24.63	**53.80**	16.83	2.10
Level 1	0	3	13	53	**27**	4	0.00	0.27	10.90	49.80	**32.93**	6.10
							Scenario 2					
Level 3	6	6	16	36	**28**	7	4.75	4.90	13.73	34.70	**31.08**	10.85
Level 2	0	1	6	30	**41**	22	0.00	0.30	5.05	25.85	**41.60**	27.20
Level 1	0	1	5	27	37	**30**	0.00	0.03	2.38	18.05	39.93	**39.63**
							Scenario 3					
Level 3	**80**	16	4	0	0	0	**76.15**	18.00	5.40	0.40	0.05	0.00
Level 2	10	**43**	36	10	1	0	7.55	**40.35**	39.15	11.90	0.98	0.08
Level 1	0	23	41	**31**	4	0	0.00	16.88	42.83	**34.43**	5.45	0.43
							Scenario 4					
Level 3	**86**	11	3	0	0	0	**83.83**	11.58	4.35	0.25	0.00	0.00
Level 2	5	29	**44**	19	3	0	3.88	23.93	**45.73**	22.83	3.23	0.43
Level 1	0	10	26	48	14	**2**	0.00	5.25	23.53	51.95	15.90	**3.38**
							Scenario 5					
Level 4	**83**	15	2	0			**81.13**	15.30	3.30	0.28		
Level 3	49	**37**	14	1			45.88	**34.35**	16.95	2.83		
Level 2	1	31	**48**	20			0.55	25.55	**49.60**	24.30		
Level 1	0	16	40	**44**			0.00	6.30	43.35	**50.35**		

Table 5. Percentages of the maximum-tolerated dose combination recommendation. Percentages at the maximum-tolerated combinations (or combinations closest to them) are in bold

The following call of the SAS macro runs a simulation study for the scenario 1 (Table 1, page 219, Wang).

```
%TDDD( max_level1=3,
       max_level2=6,
       n_patient=54,
       gamma=0.20,
       _a_=0.05/0.1/0.2/0.3/0.5/0.7,
       _b_=0.05/0.1/0.2,
       start_up=1,
       scenario=    0.03/0.05/0.08/0.13/0.20/0.29/
                    0.05/0.08/0.13/0.20/0.29/0.40/
                    0.08/0.13/0.20/0.29/0.40/0.53,
       g_size1=2,
       group_size=3,
       nb_simul=4000);
```

The macro variables are: max_level1: number of dose levels of second agent t $(t_j, j = 1, \ldots, J)$, max_level2: number of dose levels of first agent s $(s_i, i = 1, \ldots, I)$, n_patient: total number of patients, gamma: the target probability of toxicity, _a_: working model for first agent s, _b_: working model for second agent t, start_up: indicates if a startup phase was initiated (1) or not (0), scenario: defined a matrix of true toxicity probabilities where rows and columns represent dose levels of second agent t $(t_j, j = 1, \ldots, J)$ and first agent s $(s_i, i = 1, \ldots, I)$ respectively, g_size1: number of patients per cohort in the start-up, group_size: number of patients in the two-dimensional design, nb_simul: number of runs for each scenario.

```
options nonotes=0;

%macro estimate(in=,nbi=,nmc=,seed=);
proc mcmc data=&in seed=&seed nbi=&nbi nmc=&nmc plots=none outpost=_out
monitor=(p_post);
   array a(&max_level2);
   array b(&max_level1);

   parms alpha 1;
   parms beta 1;

   beginprior;
            array p_post(&max_level1,&max_level2);
            do _j_=1 to &max_level1;
            do _i_=1 to &max_level2;
                  p_post[_j_,_i_]=1-((1-a[_i_])**alpha)*((1-b[_j_])**beta);
            end;
        end;
   endprior;

   prior alpha beta ~ expon(iscale = 1);

   /* As for a logistic regression y=0 (no toxicity) with probability=1-pij and y=1
with probability pij */
   llike=log((1-((1-a[si])**alpha)*((1-b[tj])**beta))**y)
        +log((((1-a[si])**alpha)*((1-b[tj])**beta))**(1-y));
   model general(llike);
run;

ods output summary=sum(keep=p_post:);
proc means data=_out mean;
run;
ods output close;
proc datasets nolist;
   delete _out;
run; quit;
%mend estimate;
```

```
%macro define_dose(in=,mode=);

data &in;
       row=1;
       set &in end=eof;

       array a(&max_level2);
       array b(&max_level1);
       array prob(&max_level1,&max_level2);
       array diff(&max_level1,&max_level2);

       if eof then do;

       set sum point=row;

       %let k=0;
       %do _j_=1 %to &max_level1;
              %do _i_=1 %to &max_level2;
                     %let k=%eval(&k+1);
                 prob(&_j_,&_i_)=p_post&k._mean;
                     diff(&_j_,&_i_)=abs(prob(&_j_,&_i_)-&gamma);
              %end;
       %end;

       * FUNCTION;
       %if &mode ne 2dim_design %then %do;
              %do j=1 %to &max_level1;
                     init=diff(&j,1);indice&j=1;
                     %let i=2;
                     %do %while (&i <=&max_level2);
                            if diff(&j,&i) < init then
do;init=diff(&j,&i);indice&j=&i;end;
                            %let i=%eval(&i+1);
                     %end;
              %end;
              drop init;
       %end;
       end;
run;
/*-------------------------------------------------------------------*/
%if &mode=startup %then %do;
              /* the smallest level of agent 2 is choosed */
              %let current_tj=1;
              data _null_;
                     set &in end=eof;
                     if eof then call symput("current_si",indice1);
              run;
%end;
/*-------------------------------------------------------------------*/
%if &mode=2dim_design %then %do;
              data &in;
                     set &in end=eof;
                     array diff(&max_level1,&max_level2);
                     array near(&max_level1,&max_level2);
                     array search(&max_level1,&max_level2);

                     if eof then do;
                     %do j=1 %to &max_level1;
                            %do i=1 %to &max_level2;
                                   near(&j,&i)=99999; /* arbitrary value */
                            %end;
                     %end;
```

```
                    near(%eval(&current_tj),&current_si)=0;
                    %if %eval(&current_tj-1) ge 1   %then %do;near(%eval(&current_tj-
1),&current_si)=0;%end;
                    %if %eval(&current_tj+1) le &max_level1 %then
%do;near(%eval(&current_tj+1),&current_si)=0;%end;
                    %if %eval(&current_si-1) ge 1 %then
%do;near(&current_tj,%eval(&current_si-1))=0;%end;
                    %if %eval(&current_tj+1) le &max_level1 and %eval(&current_si-1)
ge 1 %then %do;near(%eval(&current_tj+1),%eval(&current_si-1))=0;%end;
                    %if %eval(&current_si+1) le &max_level2 %then
%do;near(&current_tj,%eval(&current_si+1))=0;%end;
                    %if %eval(&current_tj-1) ge 1 and %eval(&current_si+1) le
&max_level2 %then %do;near(%eval(&current_tj-1),%eval(&current_si+1))=0;%end;

                    %do j=1 %to &max_level1;
                           %do i=1 %to &max_level2;
                                  search(&j,&i)=diff(&j,&i)+near(&j,&i);
                           %end;
                    %end;
                    * FUNCTION;
                    init=99999; /* arbitrary value */
                    %do j=1 %to &max_level1;
                           %do i=1 %to &max_level2;
                                  if search(&j,&i) < init then
do;init=search(&j,&i);indj=&j;indi=&i;end;
                           %end;
                    %end;
                    call symput("current_tj",indj);
                    call symput("current_si",indi);
                    end;
                    drop init;
             run;
%end;
/*-----------------------------------------------------------------*/
%if &mode=recommend_dose %then %do;
             data _null_;
                    set &in end=eof;
                    if eof then do;
                           %do j=1 %to &max_level1;
                                  %global recom_dose&j;
                                  call symput("recom_dose&j",indice&j);
                           %end;
                    end;
             run;
%end;

%mend define_dose;

%macro report(in=,var=);

ods output summary=mean(keep=&var:);
ods html close;
proc means data=&in mean;
       var %do l=1 %to %sysevalf(&max_level1*&max_level2);
                           &var.&l
             %end;;
run;
ods html;
ods output close;

data mean_&var.;
       set mean;
```

```
       %let k=0;
       %do j=1 %to &max_level1;
              %do i=1 %to &max_level2;
                     %let k=%eval(&k+1);
                     agentB=&j;
                     %if &var=n_pat %then
%do;agentA_level&i=((&var.&k._mean/&n_patient)*100);%end;
            %if &var=n_tox %then %do;agentA_level&i=&var.&k._mean;%end;
                     %if &var=recommended_dose %then %do;
agentA_level&i=&var.&k._mean*100;%end;
              %end;
              output;
       %end;
       keep agentB  %do i=1 %to &max_level2; agentA_level&i %end;;
run;

data mean_&var;
       length characteristic $15;
       set mean_&var;
       if _n_=1 then characteristic="&var";
run;

%mend report;

%macro TDDD(max_level1=,
                     max_level2=,
                     n_patient=,
                     gamma=,
                     _a_=,
                     _b_=,
                     start_up=,
                     scenario=,
                     g_size1=,
                     group_size=,
                     nb_simul=);

%put %str(&scenario);
%if &max_level2 < &max_level1 %then %put "Error: the number of dose for the first
agent is lower than those for the second agent";
%let init_time = %sysfunc(time());%put Init time: &init_time;

data pool;
       if (0);
run;

/*-----------------------------------------------------------------*/
/* Starting of simulation ------------------------------------ */
%do rep=1 %to &nb_simul;

    ods _all_ close;

    %put SIMULATION: &rep;

/* Important to define the current level of agent 1 (current_si) and agent 2
(current_tj) */
       /* for the 2 design escalation */
       %global current_tj current_si hist end_2_design;
       %let end_2_design=0;

/*-----------------------------------------------------------------*/
/* Start-up -------------------------------------------------------*/
/*-----------------------------------------------------------------*/
       %if &start_up=0 %then %do;
```

```
              data startup;
                  /* Scenario_i_j contains the probability of true toxicity with J (t1,
...,tJ) row and I (s1,...,sI) column */
                    array scenario(&max_level1,&max_level2);
                    %let ind=0;
              %do _j_=1 %to &max_level1;
                            %do _i_=1 %to &max_level2;
                            %let ind=%eval(&ind+1);
                                  scenario(&_j_,&_i_)=%scan(&scenario,&ind,"/");
                            %end;
                    %end;
                    /* Working model ai and bj */
                    array a(&max_level2);
                    array b(&max_level1);

                    %do _i_=1 %to &max_level2;
                          a(&_i_)=%scan(&_a_,&_i_,"/");
                    %end;

                    %do _j_=1 %to &max_level1;
                          b(&_j_)=%scan(&_b_,&_j_,"/");
                    %end;
              run;
              %let current_tj=1;%let current_si=1;%let nobs=0;
       %end;
/*---------------------------------------------------------------------*/
       %else %if &start_up=1 %then %do;

              data startup/*(drop=p y sum stop end_level tox l)*/;
                    /* Scenario_i_j contains the probability of true toxicity with J
(t1, ...,tJ) row and I (s1,...,sI) column */
                    array scenario(&max_level1,&max_level2);
                    %let ind=0;
              %do _j_=1 %to &max_level1;
                            %do _i_=1 %to &max_level2;
                            %let ind=%eval(&ind+1);
                                  scenario(&_j_,&_i_)=%scan(&scenario,&ind,"/");
                            %end;
                    %end;
                    /* Working model ai and bj */
                    array a(&max_level2);
                    array b(&max_level1);

                    %do _i_=1 %to &max_level2;
                          a(&_i_)=%scan(&_a_,&_i_,"/");
                    %end;

                    %do _j_=1 %to &max_level1;
                          b(&_j_)=%scan(&_b_,&_j_,"/");
                    %end;

                    /* Initialisation of toxicity probabiliy, patient number and
toxicity number */
                    %let size=%sysevalf(&max_level1*&max_level2);
                    array n_pat(&max_level1,&max_level2);   retain n_pat1-n_pat&size
0;
                    array n_tox(&max_level1,&max_level2);   retain n_tox1-n_tox&size
0;

                    /* Initialisation of tj agent 2 and si agent 1 */
                    tj=1;si=1;
```

```
                    /* Use to assign an idenfifiant for each observation and compute
the number of toxicities for each cohort */
                    retain patid sum 0;
                    stop=0;

                    /* Seed for the generation of bernoulli variable */
                    call streaminit(&rep);
                    do while (tj<=&max_level1 and stop=0);
                           end_level=0;tox=0;
                           do while (si<=&max_level2 and end_level=0 and tox=0);

                                  do l=1 to &g_size1; /* Cohort of g_size1 patients */
                                         patid+1;
                                         p=scenario(tj,si);
                           y=rand("bernoulli",p);
                                         sum+y; /* Allow to compute the total number of
toxicities */
                                         n_pat(tj,si)+1;
                                         n_tox(tj,si)+y;
                                         output;
                                  end;
                                  if sum ge 1 then do; /* We observe at least one
toxicity. So the start-up is terminated at the current level of agent 2 */
                                                       tox=1;
                                                       if si ge 3 then si=si-2;
                                                       else stop=1; /* Stop because
early toxicity but not end of level si */
                                  end;
                                  else if sum=0 then do;
                                                if si ne &max_level2 then si=si+1; /*
Escalation of first agent i+1 */
                                                else if si=&max_level2 then
do;end_level=1;si=si-2;end; /* End of level first agent with no toxicity */

                                  end;
                                  sum=0; /* Useful to set at 0 the number of toxicities
for the next cohort */
                           end;
                           tj+1;
                    end;
             run;
/*-----------------------------------------------------------------*/
/* Estimate the parameters and the probability of toxicity on data on the start-up */
             %estimate(in=startup,nbi=1000,nmc=5000,seed=1);
             %define_dose(in=startup,mode=startup);

             proc sql noprint;
                           select count(*) into: nobs
                           from startup;
             quit;
      %end;
/*-----------------------------------------------------------------*/
      /* End of start-up--> 1: acquire some data and dose level chosen for the */
      /*starting combination for the 2 dimensional design */

      data simul&rep;
             set startup;
      run;
      /*proc datasets lib=work nolist;
             delete startup;
      run;
      quit;*/
```

```
/*-------------------------------------------------------------*/
/* dose escalation -----------------------------------------------*/;

    %let  group_size__=&group_size;
    %do %while (&nobs lt &n_patient);
        %if %eval(&nobs+&group_size__) >&n_patient %then %do;%let
group_size__=%eval(&n_patient-&nobs);%end;
             data simul&rep;

                    retain patid;
                    array scenario(&max_level1,&max_level2);

            %let size=%sysevalf(&max_level1*&max_level2);
                    array n_pat(&max_level1,&max_level2); retain n_pat1-n_pat&size;
                    array n_tox(&max_level1,&max_level2); retain n_tox1-n_tox&size;

                    set simul&rep end=eof;
            output;

                    if eof then do;
                    /* seulement pour mettre à . les lignes et remplir que la dernière
ligne de chaque cohorte*/
                    /* on supprimera à la fin peut-être */
                    array prob(&max_level1,&max_level2);array
diff(&max_level1,&max_level2);array search(&max_level1,&max_level2);
                    %do _j_=1 %to &max_level1;
                           %do _i_=1 %to &max_level2;
                                  prob(&_j_,&_i_)=.;
                                  diff(&_j_,&_i_)=.;
                                  search(&_j_,&_i_)=.;
                           %end;
                           indice&_j_=.;
                    %end;

                           call streaminit(%eval(&rep+&nobs));
                           do l=1 to &group_size__; /* Cohort of n_design patients for
2 dimensional design */
                                                  patid+1;
                                                  tj=&current_tj;si=&current_si;
                                                  p=scenario(tj,si);
                                      y=rand("bernoulli",p);
                                                  n_pat(tj,si)+1;
                                                  n_tox(tj,si)+y;
                                                  output;
                           end;
                    end;
             run;

             %if  &start_up=0 %then %do;
             data simul&rep;
                    set simul&rep;
                    if patid=. then delete;
             run;
             %end;

             /* For the final analysis, we recommend one level of agent 1 by level of
agent 2 */
             %let nobs=%eval(&nobs+&group_size__);

             %if  (&nobs lt &n_patient) %then %do;
                    %estimate(in=simul&rep,nbi=1000,nmc=5000,seed=1);
                    %define_dose(in=simul&rep,mode=2dim_design);
             %end;
```

```
		%else %do;
			%estimate(in=simul&rep,nbi=1000,nmc=5000,seed=1);
			%define_dose(in=simul&rep,mode=recommend_dose);
		%end;

	%end;

	data simul&rep;
		set simul&rep end=eof;

		array recommended_dose(&max_level1,&max_level2);

		if eof then do;
		%do j=1 %to &max_level1;
			%do i=1 %to &max_level2;
				recommended_dose(&j,&i)=0;
			%end;
		%end;
		end;
		if eof then do;
			%do j=1 %to &max_level1;
				recommended_dose(&j,&&recom_dose&j)=1;
			%end;
		end;
	run;

data pool;
	set pool simul&rep(in=a);
	if a then simul=&rep;
run;

proc datasets lib=work noprint;
	delete startup postSumInt simul&rep;
run;
quit;

ods listing;
dm 'ODSRESULTS' clear editor;
%end;
/*-------------------------------------------------------------*/
/* End of simulation ---------------------------------------*/

/*-------------------------------------------------------------*/
/* Summary of results --------------------------------------*/

proc sort data=pool;by simul;run;

data summary;
	set pool(keep=simul n_pat: n_tox: recommended_dose:);
	by simul;
	if last.simul;
run;

%report(in=summary,var=n_pat);%report(in=summary,var=n_tox);%report(in=summary,var=rec
ommended_dose);

data empty_row;
run;

data results_final;
	set mean_n_pat empty_row mean_n_tox empty_row mean_recommended_dose;
run;
```

```
proc print data=results_final noobs;
run;
proc datasets lib=work noprint;
      delete pool summary empty_row mean mean_n_pat mean_n_tox mean_recommended_dose;
run;
quit;

%let end_time = %sysfunc(time());%put End Time: &end_time;
%let duration = %sysevalf((&end_time.-&init_time.)/60);
%put Execution time : &duration.;

%mend TDDD;
```

We can easily extend this SAS macro for implementing a 3-parameter model (including the interaction term) and integrate the principle of power prior developed by Ibrahim (Ibrahim).

CONCLUSION

We illustrated the SAS v9.4 MCMC procedure through 2 applications in phase 1 cancer clinical trial. The first estimates the model parameters of CRM design and the second implements a dose-escalation method for dual agent. This offers new SAS tools for all statisticians involved in early oncology trials.

REFERENCES

O'Quigley et al.1990. Continual Reassessment method: a practical design for phase I clinical trials in cancer. Biometrics. 46:33-48

Ishizuka N. 2001. An Application of Continual Reassessment Method Using SAS System. PharmaSUG, SP13, 2001 Available at http://www.lexjansen.com/pharmasug/2001/Proceed/Stats/SP13_ishizuka.pdf

Hirakawa A et al. 2015. A comparative study of adaptive dose-finding designs for phase I oncology trials of combination therapies. Statistics in Medicine. 34(24):3194-213

Riviere MK et al. 2015. Competing designs for drug combination in phase I dose-finding clinical trials. Statistics in Medicine. 34(1):1-12

Wang K, Ivanova A. 2005. Two-dimensional dose finding in discrete dose space. Biometrics.61:217-222

Chen F. 2008. Bayesian Modeling Using the MCMC Procedure. Proceedings of the SAS Global Forum 2008 Conference. Available at http://support.sas.com/resources/papers/proceedings09/257-2009.pdf

Cheung Y. K. Dose finding by the continual reassessment method. Chapman & Hall/CRC Biostatistics Series.

Ibrahim JG et al. 2015. The power prior: theory and applications. Statistics in Medicine.34(28):3724-49

CONTACT INFORMATION

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

Name: Gwénaël Le Teuff
Enterprise: Gustave Roussy
Address: Biostatistics and Epidemiology unit, INSERM U1018 (CES-P) / OncoStat, Gustave Roussy Cancer Campus, 114, rue Edouard Vaillant, Paris, France
City, State ZIP: Villejuif, 94 805
Work Phone: +33 (0)1.42.11.49.55
Fax: +33 (0)1.42.11.52.58
E-mail: Gwenael.LETEUFF@gustaveroussy.fr
Web:
Twitter: