Paper 267-27

Generating Randomization Schedules Using SAS® Programming

Chunqin Deng and Julia Graz, PPD, Inc., Research Triangle Park, North Carolina

ABSTRACT

Randomization as a method of experimental control has been extensively used in clinical trials. Generating the randomization schedule has been an essential part of some trials. We discuss how to use SAS programming and SAS PROC PLAN to generate the randomization schedule. The issues related to randomization are also discussed. Case studies are used to illustrate the use of SAS programming and SAS PROC PLAN in generating randomization schedules.

Keyword: Randomization Schedule, Clinical Trial, PROC PLAN

INTRODUCTION

Randomization as a method of experimental control has been extensively used in clinical trials. The fundamental occasion for randomization is the assignment of treatment conditions to patients or, as preferred herein, assignment of patients to treatment conditions. Randomization tends to produce treatment groups in which the distributions of prognostic factors, known and unknown, are similar. In combination with blinding, randomization helps to avoid possible bias in the selection and allocation of patients arising from the predictability of treatment assignments. From a statistical standpoint, randomization provides for unbiased estimates of error variance and for independence of errors.

In practice, randomization requires generating the randomization schedules which should be reproducible. Generation of a randomization schedule usually includes obtaining random numbers and assigning random numbers to patients or treatment conditions. Random numbers can be generated by computer or can come from a random number table found in most statistics textbooks. For some simple randomized controlled trials with small sample sizes, randomization can be done easily by assigning random numbers from a random number table to the treatment conditions and translate the random number into the treatment assignment.

However, if the sample sizes are large, or if a restricted randomization or stratified randomization will be performed, or if an unbalanced treatment allocation ratio will be used, it is better for us to use computer programming to do the randomization.

UNRESTRICTED RANDOMIZATION, BLOCK RANDOMIZATION, AND STRATIFIED RANDOMIZATION

The most elementary form of randomization is unrestricted randomization in which treatments are allocated to patients in sequence based on random number without any restriction. Although unrestricted randomization is an acceptable approach, some advantages can generally be gained by randomizing patients in blocks, which is usually called block randomization or restricted randomization. Block randomization helps to increase the comparability of the treatment groups, particularly when patient characteristics may change over time, as a result, for example, of changes in recruitment policy. It also provides a better guarantee that the treatment groups will be of nearly equal size. In crossover trials it provides the means of obtaining balanced designs with their greater efficiency and easier interpretation. Care should be taken to choose block lengths that are sufficiently short to limit possible imbalance, but that are long enough to avoid predictability towards the end of the sequence in a block.

Another randomization approach used in practice is stratified randomization, in which separate randomization lists are produced for each subgroup or stratum. It is usually used for controlling the imbalance of baseline characteristics such as gender, age group, or baseline condition. In multi-center clinical trials, randomization can be conducted centrally or conducted at each center. This can be looked at as an example of stratified randomization with center as a stratum effect. Stratified randomization can also be used in dose escalation clinical trials where we randomize the patients within each dose cohort.

TREATMENT / PLACEBO RATIO

Clinical trial designs frequently use placebo group as a control. The ratio of treatment to placebo could be 1:1 (balanced design) and x:1 (x>1, unbalanced design). For example, in a pain study, we may compare the analgesic effects of an investigational new drug (IND), an active control, and placebo. Since we would expect no analgesic effect from placebo, we can minimize the number of patients in the placebo group. An unbalanced treatment / placebo ratio can be used: IND:Active:Placebo = 2:2:1. When an unbalanced treatment / placebo ratio is required for randomization, manual randomization would be time-consuming and error-prone. However, the randomization can be easily implemented using programming.

WHEN TO RANDOMIZE THE PATIENTS

When we generate a randomization schedule, we also need to know when (or at what stage) the patient will be randomized. A special situation is the Zelen design. According to the Zelen design, randomization precedes the patients' informed consent, which is only sought from those allocated to the experimental arm of a trial. The control group is thus unaware that randomization has taken place. As a controversial method, the Zelen design has been often suggested but rarely used. In practice, a patient is usually randomized after the

informed consent is obtained and after the patient meets the inclusion/exclusion criteria of the study.

SAS PROGRAMMING FOR RANDOMIZATION

SAS programming can be used to generate randomization schedules of complexity. The process includes using SAS programming to generate random numbers and using the SAS procedure PROC PLAN to generate the randomization schedule.

RANDOM NUMBER GENERATING

To generate a randomization schedule, the first thing we have to do is to generate random numbers. Base SAS provides numerous random number generators including the RANUNI function, which returns a random value from a uniform distribution. The RANUNI function uses a prime modulus multiplicative congruential generator with modulus (2^{31} - 1) and multiplier 397204094 that has been well known for over 35 years. PROC PLAN uses the same random number generator as RANUNI function.

The random numbers generated by the generator above are pseudo random. By using the same seed, we can get the same random number. This provides us the possibility of reproducing a randomization schedule.

For example, the following program will produce 100 four digit random numbers.

```
data randnum;
  seed=date();
  do i=1 to 100;
  x1=int(ranuni(seed)*10000);
  x=put(x1, z4.);
  output;
  end;
  drop seed x1;
run;
proc print data=randnum;
id i;
run;
```

The following random numbers will be generated: 1783, 2110, 9638, 0172, 7250, 0373, 8595, 5026, 1095, 7645, …… By assigning these random numbers to treatments, the randomization can be realized.

GENERATING A RANDOMIZATION SCHEDULE USING PROC PLAN

The PLAN procedure constructs designs and randomizes plans for factorial experiments, specifically nested and crossed experiments and randomized block designs. PROC PLAN can also be used for generating lists of permutations and combinations of numbers. The PLAN procedure can construct the following types of experimental designs:

- full factorials, with and without randomization
- certain balanced and partially balanced incomplete block designs
- generalized cyclic incomplete block designs
- Latin square designs

PROC PLAN generates designs by first generating a selection of the levels for the first factor. Then, for the second factor, PROC PLAN generates a selection of its levels for each level of the first factor. In general, for a given factor, the PLAN procedure generates a selection of its levels for all combinations of levels for the factors that precede it.

PROC PLAN is a relatively simple procedure. It contains the following four statements.

- PROC PLAN Statement: starts the PLAN procedure and, optionally, specifies a random number seed
- FACTORS Statement: The FACTORS statement specifies the factors of the plan and generates the plan
- OUTPUT Statement: saves the last plan generated to the specified data set.
- TREATMENTS Statement: The TREATMENTS statement specifies the treatments of the plan to generate, but it does not generate a plan.

CASE STUDIES

We will use four examples to illustrate how to use PROC PLAN to generate randomization schedules for various clinical trial designs.

CASE STUDY 1: A THREE-ARM, PARALEL DESIGN, PLACEBO CONTROLLED STUDY

Parallel designs are often used in phase II – phase IV clinical trials to compare the efficacy of a new drug to an active control group or placebo group. In our example, we assume that we need to generate a randomization schedule for a study with three arms (a new drug treatment, an active control treatment, and a placebo group). Suppose the sample size in the study is 360. We can write a program to generate the randomization schedule. In practice, block randomization is usually used in this type of study. Before we generate the randomization schedule, we need to decide the block size. Based on the block size and the sample size, we can calculate the number of blocks. In this example, assuming a block size = 12, the number of blocks will be 360/12 = 30.

```
options ls=132 ps=60;
title1 "RANDOMIZATION SCHEDULE";
title2 "A Randomized, Three Arm, Parallel
Design, Placebo-Controlled Study";
TITLE3;
Proc format;
  value treat  1='New Drug'
               2='Active Control'
               3='Placebo';
run;

proc plan seed=6457149;
```

```
 factors block=30 random treat=12
random/noprint;
 output out=first
    treat nvals=(1 1 1 1 2 2 2 2 3 3 3 3)
random;
run;

data first(keep=pid block treat);
 set first;
 pid=put(_n_, z3.);
run;

proc sort; by pid;
run;

proc print noobs uniform split='*';
 var pid treat;
 label pid="PATIENT*ID"
       treat="TREATMENT*GROUP";
 format treat treat.;
run;
```

The first 24 randomization codes are listed below. As we can see, the treatment assignments are balanced within each block. In this example, each block contains 12 patients with 4 patients for new drug treatment, 4 patients for active control, and 4 patients for placebo group.

SUBJECT ID	TREATMENT GROUP
001	New Drug
002	Active Control
003	Placebo
004	Placebo
005	Placebo
006	Active Control
007	Active Control
008	Placebo
009	New Drug
010	New Drug
011	Active Control
012	New Drug
013	New Drug
014	Placebo
015	Placebo
016	Active Control
017	New Drug
018	Active Control
019	Placebo
020	Active Control
021	Placebo
022	New Drug
023	New Drug
024	Active Control

CASE STUDY 2: A TWO-TREATMENT, TWO-PERIOD, TWO-SEQUENCE CROSS-OVER DESIGN

Cross-over designs are often used in clinical trials as an efficient way to minimize variability by using a patient as his/her own control. In our two-treatment example, patients will receive both treatments in random order. In this example, suppose we need to create a randomization schedule for 40 patients. We choose block length=8, and then the number of blocks we need for 40 patients is 40/8=5.

The following program uses PROC PLAN to generate the randomization schedule for 40 patients. Notice that each block contains 4 patients with treatment A followed by treatment B; 4 patients with treatment B followed by treatment A. In other words, the number of patients for treatment AB and BA is balanced within each block.

```
title1 "RANDOMIZATION SCHEDULE";
title2 "A Single-Dose, Randomized, Two-
period Crossover Study";
proc plan seed=122700;
 factors block=5 random sequence=8
random/noprint;
 output out=first
    sequence nvals=(1 1 1 1 2 2 2 2)
random;
run;
data first(keep=pid sequence period1
period2);
 set first;
 length period1 period2 $35;
 pid=put(_n_, z3.);
 if sequence=1 then do;
 period1='A';    period2='B';
 end;
if sequence=2 then do;
 period1='B';    period2='A';
 end;
run;
proc print data=first noobs double
uniform split='*';
 var pid sequence period1 period2;
 label pid="PATIENT*ID"
       period1="PERIOD 1"
       period2="PERIOD 2";
run;
```

The first 16 randomization codes are listed below:

SUBJECT ID	SEQ	PERIOD 1	PERIOD 2
001	1	A	B
002	2	B	A
003	2	B	A
004	1	A	B
005	2	B	A
006	1	A	B
007	2	B	A
008	1	A	B
009	1	A	B
010	1	A	B
011	1	A	B
012	2	B	A
013	2	B	A
014	2	B	A
015	1	A	B
016	2	B	A

CASE STUDY 3: A RANDOMIZED, DOUBLE-BLIND, PLACEBO-CONTROLLED, DOSE-ESCALATED, GENDER-MATCHED STUDY

Case study 3 is a phase I clinical trial designed to determine the Maximum Tolerated Dose (MTD) of a new drug. In a typical phase I trial with this objective, successive groups of three to six patients, called cohorts, are given the drug. The study starts at very low dose that is defined based on data from preclinical testing. If that dose is safe, another cohort of patients will be treated at a higher dose, with escalation continuing until a maximum tolerated dose (MTD) is defined. Various escalation schema with successively smaller increases in dose are used, with rapid escalation at lower, presumably more tolerable, doses and slower escalation as the dose increases.

In this example, the study is designed to search for the maximum tolerated dose (MTD), using dose escalation. The study is also placebo-controlled (with treatment/placebo ratio = 2:1) and gender matched (with male : female = 1:1). The study will include up to 7 cohorts. In each cohort, we should have 4 placebo (2 males and 2 females) and 8 treatment (4 males and 4 females) allocations.

The study design can be summarized in the following table.

	Sample Size				
Cohort	Placebo	Dose 1	Dose …	Dose 6	Dose 7
1	4	8			
2	4				
3	4				
4	4				
5	4		…		
6	4			8	
7	4				8
Total	28	8	8	8	8

The following program can be used to create the randomization schedule for 84 patients required by the design above.

```
options ls=132 ps=58 nodate nonumber;
title1 "RANDOMIZATION SCHEDULE";
title2 "A Randomized, Double-Blind,
Placebo-Controlled, Dose-Escalated,
Ascending Single Dose Study";
%let seed=9683473;
data rannum(keep=seed);
  do i=1 to 7;
   rannum=ranuni(&seed);
   seed=int(rannum*1000000);
   output;
  end;
run;
proc sql noprint;
  select seed
    into :seed1 - :seed7
    from rannum;
quit;

proc format;
  value sex   1='Male'
              2='Female';
  value treat 1='Placebo'
              2='dose 1'
              3='dose 2'
              4='dose 3'
              5='dose 4'
              6='dose 5'
              7='dose 6'
              8='dose 7';
run;
proc plan seed=&seed1;
  factors sex=2  treat=6 /noprint;
  output out=data1
         sex nvals=(1 2) random
         treat nvals=(1 1 2 2 2 2)
random;
run;
proc plan seed=&seed2;
  factors  sex=2  treat=6 /noprint;
  output out=data2
         sex nvals=(1 2) random
         treat nvals=(1 1 3 3 3 3)
random;
run;
......
proc plan seed=&seed6;
  factors  sex=2 treat=6 /noprint;
  output out=data6
         sex nvals=(1 2) random
         treat nvals=(1 1 7 7 7 7)
random;
run;
proc plan seed=&seed7;
   factors sex=2 treat=6 /noprint;
   output out=data7
          sex nvals=(1 2) random
          treat nvals=(1 1 8 8 8 8)
        random;
run;

%macro combine(dataset);
 %do i=1 %to 7;
   %let dataset=data&i;
   proc sort data=&dataset; by sex;
   data &dataset;
     set &dataset;
     cohort=&i;
     subid=%eval(&i.00)+(_n_);
  %end;
%mend;

%combine

data combine;
  set data1 data2 data3 data4
      data5 data6 data7;
run;

proc sort; by cohort sex;
run;

proc print data=combine noobs double
uniform split='*';
   var sex subid treat;
   id cohort;
   by cohort;
   label subid="PATIENT*ID"
         Cohort="COHORT"
           sex="GENDER"
         treat="TREATMENT*ASSIGNMENT";
```

```
   format sex sex. treat treat.;
run;
```

Part of the outputs are listed below:

COHORT	GENDER	SUBJECT ID	TREATMENT ASSIGNMENT
1	Male	101	Placebo
	Male	102	dose 1
	Male	103	dose 1
	Male	104	dose 1
	Male	105	Placebo
	Male	106	dose 1
	Female	107	Placebo
	Female	108	dose 1
	Female	109	dose 1
	Female	110	dose 1
	Female	111	Placebo
	Female	112	dose 1
2	Male	201	dose 2
	Male	202	Placebo
	Male	203	dose 2
	Male	204	Placebo
	Male	205	dose 2
	Male	206	dose 2
	Female	207	Placebo
	Female	208	dose 2
	Female	209	dose 2
	Female	210	dose 2
	Female	211	dose 2
	Female	212	Placebo
.....			
7	Male	701	dose 7
	Male	702	Placebo
	Male	703	Placebo
	Male	704	dose 7
	Male	705	dose 7
	Male	706	dose 7
	Female	707	Placebo
	Female	708	dose 7
	Female	709	dose 7
	Female	710	dose 7
	Female	711	Placebo
	Female	712	dose 7

CASE STUDY 4: A WILLIAMS DESIGN TO COMPARE FOUR FORMULATIONS

The Williams design is a special case of a crossover and Latin square design. It can be used to compare three or more formulations in a clinical trial. The Williams design is balanced and requires fewer sequences and periods. As a balanced design, it satisfies the following conditions:

- Each formulation occurs only once with each patient.
- Each formulation occurs the same number of times in each period.
- The number of patients who receive formulation *i* in some period followed by formulation *j* in the next period is the same for all $i \neq j$.

Notice that a standard Latin square design does not satisfy the conditions above. However, a Williams design can be generated by re-arranging a standard Latin square design. The following program is for a 4x4 Williams design.

```
title "RANDOMIZATION SCHEDULE FOR A 4X4
WILLIAMS DESIGN";
proc format;
value trt   1='R'
            2='T1'
            3='T2'
            4='T3';
run;

*** Generating the standard Latin Square;
proc plan seed=12345;
  factors period=4 ordered seq=4 ordered
            / noprint;
  treatments tmts=4 cyclic;
  output out=g      ;
  quit;
proc sort data=g;
  by seq period;
run;
proc transpose data=g
out=g1(rename=(col1=period1 col2=period2
               col3=period3 col4=period4)
drop= name_);
  by seq;
  var tmts;
run;

*** Obtain a mirror image of the standard
Latin square;
data g2;
  set g1;
  period11=period4;
  period22=period3;
  period33=period2;
  period44=period1;
run;

*** Interlace each column of the standard
and its mirror image;
data g1(rename=(period2=period3
period11=period2 period22=period4));
  set g2 (drop=period3 period4
          period33 period44);
  subno=put(_n_,z3.);
run;

proc sort;
  by subno;
run;

proc print noobs double uniform
split='*';
  var SEQ PERIOD1 - PERIOD4;
  id subno;
  label   subno  = "PATIENT*ID"
            seq  = "SEQ";
        period1  =  "PER1"
    period2 = "PER2"
    period3 = "PER3"
    period4 = "PER4"
    ;
   format period1 trt. Period2 trt.
          period3 trt. period4 trt.;
```

```
RUN;
```

SAS output is listed below. The program can be modified to generate the Williams design for other dimensions. In practice, there are usually several patients assigned to each sequence.

PATIENT ID	SEQ	PER1	PER2	PER3	PER4
001	1	R	T3	T1	T2
002	2	T1	R	T2	T3
003	3	T2	T1	T3	R
004	4	T3	T2	R	T1

CONCLUSION

By using four case studies, we demonstrate that SAS Programming and SAS PROC PLAN is a convenient tool for generating the randomization schedule for simple to complex clinical trial designs.

REFERENCES

1. Altman, D.G. and Bland, J.M. (1999), “Treatment allocation in controlled trials: why randomize?”, *BMJ* 318, 1209.
2. Altman, D.G. and Bland, J.M. (1999). “How to randomize?”, *BMJ* 319, 703-704.
3. ICH (1998), “Statistical Principles for Clinical Trials,” *Federal Register* Vol. 63, No. 179, September 16, page 49583.
4. Roberts, C. and Torgerson, D. (1999), “Baseline imbalance in randomized controlled trials,” *BMJ* 319, 185.
5. Chow, S.C and Liu, J.P. (2000), *Design and analysis of bioavailability and bioequivalence studies,* Marcel Dekker, Inc.
6. SAS institute Inc. (1999), *SAS/Stat User’s Guide, Verion 8, Cary*, NC: SAS Institute Inc.
7. Zelen, M. (1979), “*A new design for randomized clinical trials*,” N Engl J Med, 300, 1242-1245.
8. Zelen, M. (1990), “*Randomized consent designs for clinical trials: an update*,” Stats in Med, 9, 645-656

CONTACT INFORMATION

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

Author Name: Chunqin Deng, Julia Graz
Company: PPD, Inc.
Address: 3900 Paramount Parkway
City state ZIP: Morrisville, NC 27560
Work Phone: 919-462-5173
Fax: 919-379-2693
Email: Chunqin.Deng@rtp.ppdi.com
Web: http://www.ppdi.com