Paper 173-29

Using SAS® to Match Cases for Case Control Studies

Hugh Kawabata
Michelle Tran
Patricia Hines
Bristol-Myers Squibb, Princeton, New Jersey

ABSTRACT

In many epidemiological studies subjects are matched to make the study groups comparable. While there are no methods that can guarantee comparability, individual cases are often matched on important characteristics to provide assurances that the groups are comparable.

The process of matching cases is relatively simple: (1) for each study case, control cases are matched on the major characteristics; (2) if a control matches more than one case, one control is randomly selected; (3) randomly select the desired number of controls for each case. It is step (2) that can create the most difficulties.

This poster will illustrate how this can be done using SAS, and some approaches to deal with the difficulties in handling instances where a control subject matches more than one case subject.

INTRODUCTION

In many epidemiological studies subjects are matched to make the study groups comparable. An often-used approach is to check to see that the frequency distributions in each study group are alike. Being alike in the frequency distributions of key variables would provide evidence that the groups are comparable. However, there are instances where the overall distributions could be alike but the individual cases could vary substantially. While there are no methods that can guarantee comparability, individual case matching has often been used to provide assurances that the groups are comparable.

In these studies, there are usually two groups, one representing the group being studied and a comparison or control group. Subjects from the study group will be referred to as cases or case subjects, and subjects from the control group will be referred to as controls, or control subjects.

Data from an example in the PROC SQL chapter of the SAS Procedures Guide has been adapted to provide cases for the examples used in this paper. Using the SQL procedure is simple, from a programmer's perspective; however, it does consume machine resources.

```
* Sample data, edited from a SAS example;
* Split them into two datasets for this example.;
data study control;
infile cards;
  rand_num=uniform(0);
  input id study age lwt race smoke ptd ht ui @@;
  if study=1 then output study;
  else output control;
cards;
1  0 14 135 1 0 0 0 0     101  0 14 101 3 1 1 0 0
2  0 15  98 2 0 0 0 0     102  0 15 115 3 0 0 0 1
3  0 16  95 3 0 0 0 0     103  0 16 130 3 0 0 0 0
4  1 17 103 3 0 0 0 0     104  0 17 130 3 1 1 0 1
5  0 17 122 1 1 0 0 0     105  0 17 110 1 1 0 0 0
6  0 17 113 2 0 0 0 0     106  0 17 120 1 1 0 0 0
7  0 17 113 2 0 0 0 0     107  0 17 120 2 0 0 0 0
8  0 17 119 3 0 0 0 0     108  0 17 142 2 0 0 1 0
9  0 18 100 1 1 0 0 0     109  0 18 148 3 0 0 0 0
10 0 18  90 1 1 0 0 1     110  0 18 110 2 1 1 0 0
11 1 19 150 1 0 0 0 0     111  0 19  91 1 1 1 0 1
12 0 19 115 3 0 0 0 0     112  0 19 102 1 0 0 0 0
13 0 19 235 1 1 0 1 0     113  0 19 112 1 1 0 0 1
14 1 20 120 3 0 0 0 1     114  0 20 150 1 1 0 0 0
15 0 20 103 3 0 0 0 0     115  0 20 125 3 0 0 0 1
```

```
16 0 20 169 3 0 1 0 1     116  0 20 120 2 1 0 0 0
17 0 20 141 1 0 1 0 1     117  0 20  80 3 1 0 0 1
18 0 20 121 2 1 0 0 0     118  0 20 109 3 0 0 0 0
19 0 20 127 3 0 0 0 0     119  0 20 121 1 1 1 0 1
20 0 20 120 3 0 0 0 0     120  0 20 122 2 1 0 0 0
21 0 20 158 1 0 0 0 0     121  0 20 105 3 0 0 0 0
22 1 21 108 2 1 0 0 1     122  0 21 165 1 1 0 1 0
23 0 21 124 3 0 0 0 0     123  0 21 200 2 0 0 0 0
24 0 21 185 2 1 0 0 0     124  0 21 103 3 0 0 0 0
25 0 21 160 1 0 0 0 0     125  0 21 100 3 0 1 0 0
26 0 21 115 1 0 0 0 0     126  0 21 130 1 1 0 1 0
27 0 22  95 3 0 0 1 0     127  0 22 130 1 1 0 0 0
28 0 22 158 2 0 1 0 0     128  0 22 130 1 1 1 0 1
29 1 23 130 3 0 0 0 0     129  0 23  97 3 0 0 0 1
30 0 23 128 3 0 0 0 0     130  0 23 187 2 1 0 0 0
31 0 23 119 3 0 0 0 0     131  0 23 120 3 0 0 0 0
32 0 23 115 3 1 0 0 0     132  0 23 110 1 1 1 0 0
33 0 23 190 1 0 0 0 0     133  0 23  94 3 1 0 0 0
34 1 24  90 1 1 1 0 0     134  0 24 128 2 0 1 0 0
35 0 24 115 1 0 0 0 0     135  0 24 132 3 0 0 1 0
36 0 24 110 3 0 0 0 0     136  0 24 155 1 1 1 0 0
37 0 24 115 3 0 0 0 0     137  0 24 138 1 0 0 0 0
38 0 24 110 3 0 1 0 0     138  0 24 105 2 1 0 0 0
39 1 25 118 1 1 0 0 0     139  0 25 105 3 0 1 1 0
40 0 25 120 3 0 0 0 1     140  0 25  85 3 0 0 0 1
41 0 25 155 1 0 0 0 0     141  0 25 115 3 0 0 0 0
42 0 25 125 2 0 0 0 0     142  0 25  92 1 1 0 0 0
43 0 25 140 1 0 0 0 0     143  0 25  89 3 0 1 0 0
44 0 25 241 2 0 0 1 0     144  0 25 105 3 0 1 0 0
45 1 26 113 1 1 0 0 0     145  0 26 117 1 1 1 0 0
46 0 26 168 2 1 0 0 0     146  0 26  96 3 0 0 0 0
47 0 26 133 3 1 1 0 0     147  0 26 154 3 0 1 1 0
48 0 26 160 3 0 0 0 0     148  0 26 190 1 1 0 0 0
49 0 27 124 1 1 0 0 0     149  0 27 130 2 0 0 0 1
50 0 28 120 3 0 0 0 0     150  0 28 120 3 1 1 0 1
51 0 28 130 3 0 0 0 0     151  0 28  95 1 1 0 0 0
52 0 29 135 1 0 0 0 0     152  0 29 130 1 0 0 0 1
53 0 30  95 1 1 0 0 0     153  0 30 142 1 1 1 0 0
54 0 31 215 1 1 0 0 0     154  0 31 102 1 1 1 0 0
55 0 32 121 3 0 0 0 0     155  0 32 105 1 1 0 0 0
56 0 34 170 1 0 1 0 0     156  0 34 187 2 1 0 1 0
 ;
```

The data in this example are set up so that there are two observations in each line; hence the need for the double '@' in the input statement. The data step reads the data and creates two SAS datasets out of the raw data, one containing the study cases and the other, the control cases. A random number is assigned to each record for use in the examples below.

SIMPLE MATCH: EXACT MATCH ON AGE AND RACE

In this simple example, the subjects are matched on age and race. For each case you get all control subjects that match the case's age and race. This is easily done using PROC SQL:

```
PROC SQL;
CREATE table controls_id
   as select
      one.ID as study_id,
      two.ID as control_id,
      one.age as study_age,
      two.age as control_age,
      one.race as study_race,
      two.race as control_race,
      one.rand_num as rand_num
  from study one, control two
  where (one.age=two.age and
```

```
            one.race=two.race);
```

Note that in SQL every record in CONTROL is first matched with every record in STUDY. Then, the WHERE clause is executed so that only those records that meet the criteria in the WHERE clause is kept. Since a control subject's age and race may match more than one case subject, a control subject may be duplicated in the control dataset. This can be removed by randomly choosing only one matched pair, using the random number assigned in the DATA step above.

```
* Remove duplicate control subjects;
proc sort data=controls_id nodupkey;
  by control_id rand_num;
  run;
```

One problem with this method is that some case subjects will have more control subjects than others and could produce significant bias. In a unlikely example, you may have 50 times as many control subjects for one age than the other ages, making the controls not comparable to the cases.

SIMPLE MATCH: EXACT MATCH ON AGE AND RACE WITH FIXED NUMBER OF CONTROLS

In this example, the subjects are matched as before on age and race, but only two controls are allowed for each case subject.

As before, PROC SQL is used to match the subjects, and the duplicate control subjects are removed.

```
PROC SQL;
CREATE table controls_id
   as select
      one.ID as study_id,
      two.ID as control_id,
      one.age as study_age,
      two.age as control_age,
      one.race as study_race,
      two.race as control_race,
      one.rand_num as rand_num
  from study one, control two
  where (one.age=two.age and
         one.race=two.race);
* Remove duplicate control subjects;
proc sort data=controls_id nodupkey;
  by control_id rand_num;
  run;
```

Next randomly select the fixed number (in our example, two) of control subjects for each case. Sort the data by the case subject ID's and the random number, and keep the first set of control subjects encountered.

```
proc sort data=controls_id ;
  by study_id rand_num;
  run;
data controls_id2 not_enough;
set controls_id;
  by study_id ;
  retain num;
  if first.study_id then num=1;
  if num le 2 then do;
    output controls_id2;
    num=num+1;
    end;
  if last.study_id then do;
    if num le 2 then output not_enough;
    end;
  run;
proc print data=controls_id2(obs=40);
  title2 'matched patients';
  run;
```

The SAS dataset, "not_enough," contain the study cases that do not have two controls. These cases can be removed with the following code fragment:

```
data controls_id3;
merge controls_id2
      not_enough(in=b_);
  by study_id;
  if b_ then delete;
  run;
```

MATCHING ON RANGE OF VALUES

Rather than matching exactly on age, you are more likely to match on an age interval (plus or minus one year, in this example). This can add more complexities than is first apparent.

First modify the control subjects slightly by creating the upper and lower bounds for age:

```
data control2;
set control;
  age_low=age-1;
  age_high=age+1;
  run;

proc sql;
  create table controls_id as
    select
    one.ID as study_id,
      two.ID as control_id,
      one.age as study_age,
      two.age as control_age,
      one.race as study_race,
      two.race as control_race
  from study one, control2 two
  where ((one.age between two.age_low and two.age_high)
        and one.race=two.race);
```

To remove the duplicate control subjects, you encounter a more complicated issue. To illustrate the problem, consider the following simple scenario: there are two case subjects, A and B, and four control subjects, a, b, c and d, whose age and race are listed.

case subject A, age=18, race=W

case subject B, age=20, race=W

control subject a, age=17, race=W

control subject b, age=18, race=W

control subject c, age=19, race=W

control subject d, age=21, race=W

In this situation case subject A matches three control subjects, a, b and c, and, case subject B matches two control subjects c and d. Control subject c matches both case subjects A and B. Ideally, control subjects a and b would be assigned to case subject A, and case subject B would match controls c and d.

One simple way of handling this is to order the control subjects by the number of matches they have with the case subjects. Then keep the matches for the low frequency control subjects first.

```
* count the number of control subjects for each case subject;
proc sort data=controls_id;
  by study_id;
  run;
data controls_id2(keep=study_id num_controls);
set controls_id;
  by study_id;
  retain num_controls;
  if first.study_id then num_controls=1;
```

```
  else num_controls=num_controls+1;
  if last.study_id then output;
  run;
* now merge the counts back into the dataset;
data controls_id3;
merge controls_id
      controls_id2;
  by study_id;
  run;
* now order the rows to select the first matching control;
proc sort data=controls_id3;
  by control_id num_controls rand_num;
  run;
data controls_id4;
set controls_id3;
  by control_id;
  if first.control_id;
  run;
```

Now, as before, randomly select the fixed number (in our example, two) of control subjects for each case.

```
proc sort data=controls_id ;
  by study_id rand_num;
  run;
data controls_id2 not_enough;
set controls_id;
  by study_id ;
  retain num;
  if first.study_id then num=1;
  if num le 2 then do;
    output controls_id2;
    num=num+1;
    end;
  if last.study_id then do;
    if num le 2 then output not_enough;
    end;
  run;
proc print data=controls_id2(obs=40);
  title2 'matched patients';
  run;
```

There are two problems with this approach, one technical and the other substantive. The technical problem is with ties. In the example above, the rows are randomly sorted as well, so that in the event of ties the controls are randomly assigned to the case.

In the example above, if control subject b also matched case subject B, sometimes the control would be assigned to A which would be correct, and other times it would be assigned to B, which would mean case A would have insufficient matches. While it is easy to identify the problem in this overly simple case, in real life it is difficult to identify them.

The substantive problem is with bias. The whole point in matching is to balance the important attributes of the subects between two (or more) groups so that they are comparable. If the matching process adds bias, the intent of matching is defeated. Consequently, it is important to evaluate the groups after the matching process is complete.

CHECKING THE MATCHING PROCESS

One approach to evaluating the matching process is to repeat the matching process and evaluating key criteria. In the context of a research project there will be indicators of bias that could be evaluated after each iteration of the matching process. In the absence of a research project, you can choose some criteria; we have chosen the number of total number of cases and controls as an example.

In this example code fragment, the removal of duplicates is repeated 100 times and the resulting dataset with the maximum number of rows is kept. In the context of a research project, a more substantive measure could be used for selection (for example, age).

```
%macro test;
```

```
%do ii = 1 %to 100;
data random;
set controls_id3;
  sample_seed=&ii;
  rand_num=uniform(&ii);
  run;
proc sort data=random;
  by control_id num_controls rand_num;
  run;
data controls_id4;
set random;
  by control_id;
  if first.control_id;
  run;
proc sort data=controls_id4 ;
  by study_id rand_num;
  run;
data controls_id5 not_enough;
set controls_id4;
  by study_id ;
  retain num;
  if first.study_id then num=1;
  if num le 2 then do;
    output controls_id2;
    num=num+1;
    end;
  if last.study_id then do;
    if num le 2 then output not_enough;
    end;
  run;
proc print data=controls_id5(obs=40);
  title2 'matched patients';
  run;

data sample&ii;
merge controls_id5
      not_enough(in=b_);
  by study_id;
  if b_ then delete;
  run;
%end;

data _null_  ;
retain num1-num100;
%do ii= 1 %to 100;
  set sample&ii nobs=num&ii;
  %end;
  max_subj=max(of num1-num100);
  put 'Dataset with maximum number of subjects = sample' max_subj;
  run;
%mend test;
```

CONCLUSIONS

This paper illustrated a way to use SAS to match control subjects to case subjects for research studies using the case control methods. The issues involved in the process were illustrated and sample SAS code described.

TRADEMARKS

REFERENCES

SAS Institue Inc. 1999. SAS Procedures Guide. SAS Institute Inc., Cary, NC.

ACKNOWLEDGEMENTS

The author may be contacted at:

Hugh Kawabata
Bristol-Myers Squibb
P.O. Box 4500
Princeton, NJ 08543 USA
(609) 897-3695
E-mail: hugh.kawabata@bms.com