Paper 088-2012

Optimized 1:N Case-Control Match Using SAS®

Zhiwei Wang, Center for International Blood and Marrow Transplant Research, Medical College of Wisconsin, Milwaukee, WI

ABSTRACT

In case-control match studies, when given a set of matching criteria, cases can be matched with a large number of controls and vice versa. Randomly selecting the matches might not lead to the maximum number of matched pairs. An algorithm based on SAS that optimizes a 1:N match is presented in this paper. The algorithm uses an iteration-based approach. Two calculated statistics are used to guide the selection process and to ensure that an optimal pair is selected in every iteration. The resulting data set has the maximum number of matched pairs, and the number of pairs is evenly distributed among all cases. This algorithm can be applied to any number of matching criteria. Examples of SAS code are given in each key step.

INTRODUCTION

A case-control study evaluates the difference of exposure factors between the case and control groups. Subjects with the studied condition are called cases and subjects without the condition are called controls. To ensure that the two groups of subjects are comparable, it is important to match the two groups of subjects on characteristics variables.

In simple studies, each case is matched with 1 and only 1 control. This is called a 1:1 match. In some studies (such as studies on rare diseases) where the number of cases is small, a 1:N match is often preferred as it can increase the statistical power of detecting the association. In a 1:N match, each case is matched with up to N controls.

In terms of matched variable values, there are two types of match. The exact match matches subjects on exact variable values; for example, matching subject with the exact same age in years. The range match matches subject on a specified range of values; for example, matching subjects on body weight in a range of +/- 5 pounds.

With a large number of cases and controls, picking out matched pairs in an optimized way can be difficult, especially when there is range match involved. This can be shown by the following examples.

Suppose we have two sets of subjects: data set A is the case data set and data set B is the control data set. The two sets are to be 1:1 matched on age by a range of +/- 1 year. Let's consider several scenarios of matches as shown in Figure 1.

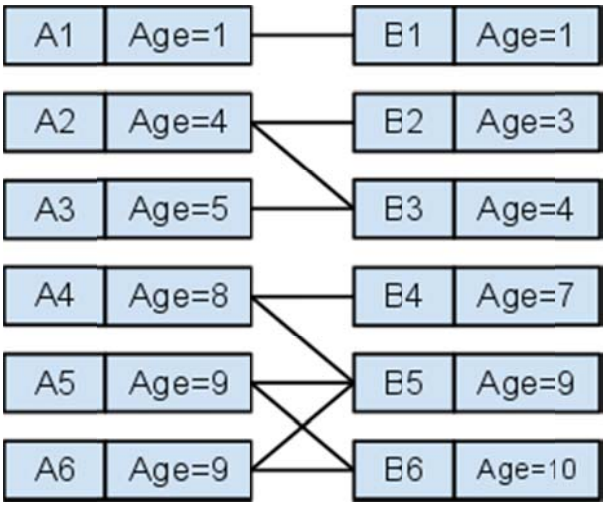

Figure 1: Case Control Matching by Age

First, A1-B1 is the simplest match because there is only one case with age of 1 to be matched with the only control of the same age.

Second, case A2 can be matched with both control B2 and B3 while case A3 can be matched with B3 only. Suppose we match A2 with B3, then A3 cannot find a match elsewhere and therefore will be excluded from the study. Subjects that cannot find matches are called orphan subjects. In the meantime, B2 remain used. This is not an optimal solution. Now suppose we match A2 with B2, then A3 can be matched with B3. This time we have two matched pairs, and no orphan subjects. The A3-B3 is a 'rarer' pair because A3 have only one choice of match and A2 has two. So strategically we would like to select the case with lower number of matches (the 'rarer' one) first.

Third, a more complicated situation with 3 cases (A4, A5, A6) and 3 controls (B4, B5, B6). A4, A5 and A6 each have two pair of matches. The strategy in the second scenario says we should pick the case with lower number of matches first. However, this strategy does not provide guidance here because each case has the same number of matches. Randomly picking out the pairs still may not give us optimal results. An example is that if A4-B5 and A6-B6 are picked as pairs, we are left with an orphan case (A5) and an orphan control (B4). To find the optimal match, let's examine the number of matches for each control: B4 has one match, B5 has three and B6 has two. This tells us that A4-B4 should be picked first because B4 cannot be matched otherwise. The rest is easy: A5 can match either B5 or B6 and A6 will match the remaining one.

The above examples showed that when matching, randomly picking out matched pairs is likely to result in orphan subjects that could otherwise be matched with. It is important to consider the number of matches each case and each control has, and assign the higher priority of the 'rarer' pairs to be picked first. Following this logic, an algorithm based on SAS is presented below. The algorithm can handle 1:N matches and the match is optimized so that each case is matched with the maximum number of available controls.

Since the process has a selection bias based on the number of matches, an important assumption is that the number of matches of each subject is random and independent of the distribution of the exposure and response variables. While uncommon but if for any reason this condition cannot be assumed, this algorithm should not be used.

THE ALGORITHM

The algorithm manipulates two input data sets, one interim data set and one output data set. The two input data sets, the case pool and the control pool, store all the subjects that have not yet been matched. The interim data set is called the potential pool and it temporarily stores all the potential matching pairs. The algorithm evaluates on the potential pool and picks out the pairs to the output data set. The output data set is called the matched pool. In the beginning, the case pool and control pool contain all the cases and controls available to the study and the matched pool is empty. The goal of the algorithm is to move cases and controls from the case pool and control pool to the matched pool.

The algorithm carries out a loop. Each iteration starts with the case and control pool. Subjects in these two pools are to be matched, and the resulting data set is the potential pool. Two statistics SCARCE and DEMAND are calculated for each case and control in the potential pool. One 'optimal' choice of matched pair is selected based on the two scores. The selected pair goes to the matched pool. The remaining unselected subjects return to the case or control pools. The looped process continues until the case pool, the control pool or the potential pool become empty. The program flow is illustrated in Figure 2. The rest of this section examines each step of the loop in details and explains the two scores.

At initialization, the case pool and control pool data sets should both contain an 'id' variable as the unique IDs for each subject, and all thevariables being matched. Each of the case and control subject should also be assigned a random number, stored in variables 'case_rand' and 'control_rand' respectively. The use of these two variables will be discussed in the latter part of this section. None of the subjects should have missing values on any variables.

At the beginning of the loop, two data sets are matched. The match of cases and controls can be done using either MERGE in the DATA step or PROC SQL procedure. Case-control match on various matching criteria using PROC SQL was demonstrated by Kawabata et. al. (Kawabata et. al. SUGI 29). A simple example of matching by age using PROC SQL can be done with the following code:

```
PROC SQL;
    CREATE TABLE potential AS
    SELECT case.id as caseid, control.id as controlid, case_rand, control_rand
    FROM case FULL JOIN control
    ON case.age=control.age;
```

All the matching criteria can be specified by the 'ON' statement. The FULL JOIN statement ensures all possible matches are retained. For range match, the 'ON' statement can be specified by an inequality expression. For example, if we need to match cases' age to controls' age within 1 year difference, the code should be:

```
PROC SQL;
    CREATE TABLE potential AS
    SELECT case.id as caseid, control.id as controlid
    FROM case FULL JOIN control
    ON -1<=case.age-control.age<=1;
RUN;
```

The resulting potential pool data set contains all the possible matched pairs that satisfy the matching criteria. Note cases and controls are duplicated in this set and obviously we cannot take them all (hence the term 'potential'). Randomly picking out any pair does not result in optimal solution, for reasons explained in the introduction section. We need the algorithm to select one 'optimal' pair at a time so that we can reach an optimal final set at the end. The definition of 'optimal' is based on the two factors: SCARCE and DEMAND. We define, for each case, SCARCE is the number of matched controls; and for each control, DEMAND is the number of matched cases. A matched pair with SCARCE=1 and DEMAND=1 means the case and control can only be matched with each other, not any other subjects. A matched pair with SCARCE=3 and DEMAND=2 means the case in this pair can be matched with 3 controls (1 in this pair and 2 in other pairs); the control in this pair can be matched with 2 cases (1 in this pair and 1 in another pair).

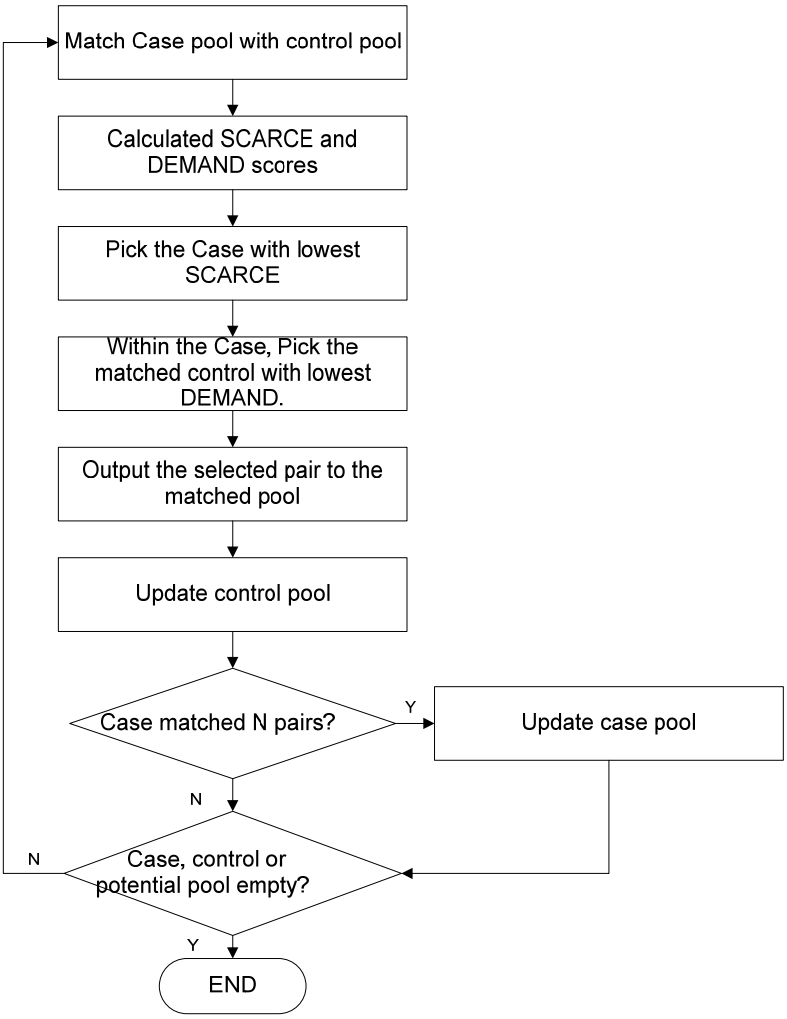

Figure 2: Algorithm Flow Chart

In SAS, SCARCE and DEMAND can be calculated using PROC SQL procedures.

```
PROC SQL;
    CREATE TABLE scarcecount AS
    SELECT caseid, count(controlid) AS scarce
    FROM potential
    GROUP BY caseid;

    CREATE TABLE demandcount AS
    SELECT controlid, count(caselid) AS demand
    FROM potential
    GROUP BY controlid;
RUN;
```

The data sets ‘scarcecount’ and ‘demandcount’ need to be merged back to the potential pool data set so that the statistics can be used in the selection.

For cases with SCARCE=1, we simply have to select the cases and their matching cases, otherwise the cases will be excluded from the study. If multiple cases with SCARCE=1 are matched with one same control, this is called a tie. The algorithm randomly picks one case out of the tied cases. Of course, the remaining cases will never be able to find other matches. This happens because the data sets do not have enough to match, and apparently there is nothing we can do at the programming stage about this situation.

Now consider cases with SCARCE>1. This means each case is matched with more than 1 control. Each time we pick one control out from all potential matches. Which control should we pick? Some controls can only be matched to the case in the current pair (DEMAND=1), while other controls can be matched with more than one cases (DEMAND>1). We should choose the controls with DEMAND=1 first because they would not interfere with other case match. Following this logic, the algorithm always chooses the controls with the lowest DEMAND first. Again, if there is a tie on the DEMAND scores, the algorithm picks out a random control within the same score.

The algorithm sorts the potential pool data set by SCARCE and DEMAND in increasing order. Each time the pair with the lowest SCARCE and DEMAND is selected. If there are multiple pairs with the same SCARCE and DEMAND scores, a random pair is selected. The one pair with the lowest random number is selected first. In SAS, this can be done by a PROC SORT procedure.

```
PROC SORT DATA=potential;
    BY scarce case_rand caseid demand control_rand;
RUN;
```

The variables ‘case_rand’ and ‘control_rand’ are the random numbers assigned to each case and control subjects at the initialization step. Assigning a random number can be done in a DATA step using function UNIFORM(). Note the third sorting variable is ‘caseid’ – the unique ID of the case subject. Although very rarely but occasionally different cases can be assigned to an exactly same random value. Inserting the ‘caseid’ variable in the sort procedure technically ensures that the comparison of the DEMAND statistics is within the pairs of the same case.

The selected pair is output to the matched pool data set. In SAS DATA steps, this is simply outputting the first observation in the sorted potential pool data set and appending it to the matched pool data set. The control from the selected pair is removed from the control pool so that it cannot be matched to other cases in the next iteration. All the other controls remain in the control pool. In a 1:N match schema, if the case has less than N matches in the matched pool, it is still eligible to match more controls. The case is removed from the case pool only when it reaches the maximum number of matches (N).

After the case and control pools are updated, a new iteration of matching begins. The SCARCE and DEMAND scores need to be re-calculated for each remaining case and control. Note the SCARCE score of each case should be the sum of number of matches in the potential pool and the number of pairs of this case that are already in the matched pool. This is because a case could appear to be having only 1 match in the potential pool while in fact there are already matched pairs in the matched pool. In order to ensure the controls are evenly distributed to all cases, we do not want this case to compete with pairs that are really having only 1 match in the potential pool and no match in the matched pool. The number of pairs in the matched pool can be calculated by the PROC SQL procedure similar to the SCARCE/DEMAND calculation and merged back to the potential pool in the next iteration.

The looped process continues until any of the three pools (case pool, control pool or potential pool) is exhausted. The resulting matched pool is a data set with all case id and control id recorded in pairs. The loop can be controlled by a SAS macro. A %DO %UNTIL statement can be used. At the end of the %DO %UNTIL statement, the number of observations in the case pool, the control pool and the potential pool are checked to determine if the loop should be terminated.

DISCUSSIONS

For the sake of simplicity, the algorithm presented in this paper does not attempt to optimize the process computationally. One short coming of the algorithm is that it only selects one matched pair to the matched pool in each iteration. In the next iteration, the potential pool has to be re-constructed and scores re-calculated. This is apparently not the most efficient use of information we already have. However, it is not hard to improve the algorithm by analyzing on how the scores of affected cases and controls will change after a pair is moved to the matched pool.

For example, one simple improvement can be done is on matched pairs with both SCARCE and DEMAND score of 1. All these pairs are 1:1 matched and do not match with subjects other the one in the pair. Since the scores of the other pairs will not change after the 1:1 pairs are selected to the match pool, all 1:1 pairs can be selected and output to the matched pool in a single iteration, rather than outputting 1 pair at each iteration.

CONCLUSIONS

This paper illustrated an algorithm to optimize a 1:N case-control match using SAS. The resulting data set contains the subject IDs for matched pairs. The algorithm ensures that each case is matched with maximum number of controls and the number of matched controls is evenly distributed among all cases.

REFERENCES

Kawabata, Hugh,Tran, Michelle, Hine, Patricia. *Using SAS® to Match Cases for Case Control Studies*, Paper 173-29, SUGI 29

CONTACT INFORMATION

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

Name: Zhiwei Wang
Enterprise: Center for International Blood and Marrow Transplant Research, Medical College of Wisconsin
Address: 9200 W. Wisconsin Avenue, C5500
City, State, ZIP: Milwaukee, WI, 53226
Email: zwang@mcw.edu