One-to-One Matching of Case/Controls Using SAS® Software

Charles John Tassoni, Baibai Chen, Clara Chu
Johns Hopkins University, Dept. of Epidemiology

ABSTRACT

In an epidemiological study it is often desirable to transform separate sets of case and control subjects into a single data set in which each case subject is paired with a distinct control. This process is called one-to-one matching. Matches are made based on how well they reduce differences between case and control on a potentially confounding variable (e.g., prior health in a drug efficacy study). This article presents an example of a one-to-one matching task and the SAS code necessary to implement it. The code consists of DATA step, SQL, and MACRO programming. One-to-one matching is done in two steps. The first step is to find all acceptable controls for each case. The second step is to find the best match for each case, or, failing that, for as many cases as possible. The second step is general enough that it can be coded into a macro that does not depend on the specifics of the example presented here.

INTRODUCTION

In epidemiological studies it is often useful to eliminate potential confounders by doing one-to-one matching of case/control groups. After one-to-one matching, statistical analyses will be more powerful– at the very least, the analysis will require one less covariate. To illustrate a situation where matching would be useful, consider a study in which one wishes to assess whether and by how much an antiretroviral therapy prolongs life for patients with HIV infection. In such studies treatment/no treatment groups are rarely selected randomly; rather, people with weaker immune systems are more likely to seek treatment. Furthermore, people with weaker immune systems at the beginning of a study are likely to end the study with weaker immune systems, and are, in fact, more likely to die. Here the state of the immune system is what epidemiologists call a potential confounder, in that it is related both to treatment exposure and to outcome. The problem can be addressed by pairing case/controls so that differences in immune system state prior to treatment are reduced.

One-to-one matching may seem at first to be a difficult coding task. However, base SAS software can handle the problem elegantly, as will be demonstrated below. For clarity's sake I will continue using the antiretroviral example. I will assume further that we begin the matching task with a data set of cases and a data set of controls. Each record of the case data set has an ID, STARTDT (date at which antiretroviral therapy began), and a TCELLN (t-cell count– a variable that is often used as a proxy for immune system state– measured on STARTDT). The set of possible controls is similar, having variables ID, DT (date at which t-cell count was measured), and TCELLN. The difference between the two data sets is that in the case data set there is a record for each ID, while in the control data set there is a record for each date at which a control ID had a t-cell measurement taken. Put another way, the case data set has ID as a primary key, while the control data set has a composite primary key of ID and DT.

FINDING POTENTIAL MATCHES

Our task is to create a data set of case/control matches given the data sets above. We begin by finding all possible matches, within prescribed restrictions, for each case ID. In this example we will restrict matches so that a control is an acceptable match for a case only when the difference between their t-cell counts is no greater than 50 cells at or around the case ID's STARTDT. "At or around" will mean that an acceptable control has had his t-cell count measured within +- 60 days of the case's STARTDT. If more than one DT for a given control ID falls within the window, we choose the record with the control DT closest to the case's STARTDT.

Here is code that finds all possible control matches for each case ID, given the restrictions above.

```
PROC SQL;
CREATE TABLE POSSMCH AS
SELECT CASE.ID AS CASEID, CONTROL.ID AS CONTRLID,
       ABS(STARTDT - DT) AS DTDIFF,
       ABS(CASE.TCELLN - CONTROL.TCELLN) AS TDIFF
FROM CASE, CONTROL
WHERE ABS(CASE.TCELLN - CONTROL.TCELLN) < 50 AND
      ABS(STARTDT - DT) <= 60
ORDER BY CASEID, CONTRLID, DTDIFF;

DATA POSSMCH;
   SET POSSMCH;
   BY CASEID CONTRLID;
   IF FIRST.CONTRLID THEN OUTPUT;
```

The SQL query creates data set POSSMCH (possible matches) from data sets CASE and CONTROL. The new data set has variables CASEID, CONTRLID, DTDIFF (difference between case's STARTDT and control's DT), and TDIFF– the difference between case and control's t-cell count. (SAS programmers who do not use SQL might consider how complicated it would be to create this data set using only DATA step programming.) After this query the data set POSSMCH may have more than one date for each CASEID CONTRLID pair. The data step following the query fixes this by processing POSSMCH by CASEID CONTRLID and outputting only the first record of each CASEID CONTRLID grouping. This serves to keep the record with the smallest DTDIFF for each CASEID CONTRLID grouping, since POSSMCH has been sorted by CASEID CONTRLID DTDIFF in the SQL query.

FINDING ONE MATCH

Up to now we have found all possible control matches for each case ID. Our goal is to end up with one control for each case ID. This goal cannot always be met. For example, it cannot be met if there are less control IDs than case IDs. But the macro that closes this paper finds good matches for most, if not all, controls under typical circumstances.

```
%MACRO MATCHUP(RESULT, POTENMCH, CASID, CTRLID, DIFF); /*1*/
%LOCAL I J;

%LET I = 0;
%DO %UNTIL  (&SQLOBS = 0);                              /*2*/
  %LET I = %EVAL(&I + 1);

  PROC SORT DATA = &POTENMCH; BY &CASID &DIFF;          /*3*/
  DATA BESTMCH;                                         /*4*/
     SET &POTENMCH;
     BY &CASID;
     IF FIRST.&CASID THEN OUTPUT;

  PROC SORT DATA = BESTMCH; BY &CTRLID &DIFF;           /*5*/
  DATA MATCH&I;
     SET BESTMCH;
     BY &CTRLID;
     IF FIRST.&CTRLID THEN OUTPUT;

  PROC SQL;                                             /*6*/
  CREATE TABLE &POTENMCH AS
  SELECT &POTENMCH..*
  FROM &POTENMCH
  WHERE &CASID NOT IN
```

```
   (SELECT &CASID
           FROM MATCH&I)
  AND &CTRLID NOT IN
   (SELECT &CTRLID
           FROM MATCH&I);
%END;

PROC DATASETS; DELETE &POTENMCH;

DATA &RESULT;                                              /*7*/
   SET
   %DO J = 1 %TO &I;
          MATCH&J
   %END;
   ;
%MEND MATCHUP;
```

A description corresponding to the numbered statements appears below:

1. The parameters are, from left to right, the name the user wishes the output data set to have, the name of the input data set containing the potential matches (this data set will be deleted in the macro), and the names of the variables holding the input data set's case ids, control ids, and difference to be reduced by one-to-one matching. In the antiretroviral example we want to reduce differences in t-cells. Thus the macro is called in the following manner: %MATCHUP(MATCHES, POSSMCH, CASEID, CONTRLID, TDIFF).

2. Go through the loop once no matter what. Continue looping until the automatic macro variable SQLOBS signals that there are no rows selected in the SQL query below (see step 6). That signal will mean that the pool of potential matches has been exhausted. Note that on each loop the macro variable I is incremented.

3. For each CASID we order matches so that the smallest differences (i.e., smallest values of DIFF) occur first.

4. The first record in a CASID grouping is now the best match for that CASID, since it has the smallest difference. Data set BESTMCH thus gives every CASID its best match.

5. Not all CASID's necessarily get to keep their best match, however. When two or more CASID's have the same CTRLID as their best match then we have to break ties. To do this we sort BESTMCH by CTRLID DIFF. Now the first record in a CTRLID group will be the one with the smallest difference for that control. Also, that first record will have a unique CASID, since BESTMCH has no more than one record per CASID. We output the first record in each CTRLID group into data set MATCH&I.

6. Here we reduce the pool of potential matches by removing from POTENMCH all records that have a CASID or a CTRLID in MATCH&I. That is, we remove from POTENMCH any records with ids that have already been matched. If there are any observations left in POTENMCH, steps 3-6 are repeated. When there are no potential matches left then it is time for the loop to stop.

7. After exiting the loop, the various MATCH&J's for J from 1 to I are set together to create the data set of one-to-one matches that is output back to the user.

AUTHORS

Authors can be contacted at 615 N. Wolfe Street, Room E7005, Baltimore, MD 21209, phone: 410-955-4320, email address: author's login + @statepi.sph.jhu.edu. Author's logins are ctassoni, baibai, and cchu.