PharmaSUG 2016 - Paper PO14

Making Greedy into Optimal! A Poor Woman's Attempt to Get Optimal Propensity Score Matching from a Greedy Matching Algorithm

Janet Grubber, VA HSR&D, Durham, NC
Carl Pieper, Duke University Medical Center, Durham, NC
Cathleen Colon-Emeric, Duke University Medical Center, Durham, NC

ABSTRACT

Propensity score matching (matching on the probability of treatment assignment (treated vs. untreated) conditional on observed baseline characteristics) is a popular method used by many observational studies to approximate, as much as possible, randomized clinical trial methodology. In the medical literature, greedy matching is the form of matching most often reported, though optimal matching is often said to be a superior method. In our real world example, our goal was to match 1 treated patient to 3 untreated controls if 3 suited controls existed; however, if fewer (1 or 2) existed, we still wanted to retain the 1:2 or 1:1 matched pair to increase our power to detect significant differences in analyses. Optimal matching was well-suited to accomplish our goal; however, our organization lacked funds to pay for the needed SAS module. Greedy matching algorithms, which were runnable using our existing SAS 9.4 modules, typically create only fixed ratios of treated:untreated control matches (e.g., for a desired 1:3 ratio, only treated patients with a full complement (3) of untreated controls are retained; those with fewer matched controls (1 to 2) get dropped from the final data set and their matched controls go back into the pool of possible controls for other treated patients). Here we share our solution; we build on an existing greedy matching macro to produce matched pairs (treated:untreated) of 1:3, 1:2, and 1:1 ratios within the same data set. Our solution adds one, but not all capabilities of optimal matching to statistics toolbox! (Appropriate for intermediate users.)

INTRODUCTION

In observational studies, investigators do not have the luxury of randomly assigning participants to treated and untreated groups. Instead, participants "come as they are", treated or untreated and, as a result, the treated and untreated groups can easily differ from each other in important ways. When investigators attempt to determine whether a treatment is associated with an outcome, the analysis is ideally performed on groups identical to each other in all measureable ways, except for their treated vs. untreated condition (for example, whether they are screened or unscreened for a condition of interest, such as osteoporosis). One method used to "make" treated and untreated groups in observational studies as similar as possible to each other is through the calculation of propensity scores for each patient. Once calculated, propensity scores, defined as the probability of being treated based on observed covariates, can be used to attempt to reduce treatment selection bias through stratification on propensity scores, inverse probability of treatment weighting, covariate adjustment using propensity scores, or matching on the propensity score.

This paper focuses on the propensity score matching method. There are several types of matching in the medical literature, including greedy and optimal matching; of these greedy matching is most often reported, though optimal matching is often considered superior. Greedy matching is a linear matching algorithm in which, once a treated:untreated match is formed, the untreated control is no longer available for inclusion in future matches and, similarly, after an apriori specified number of matched controls for a treated case has been reached, the case is also removed from the pool available for future matches. Cases for whom the specified number of controls cannot be obtained are also dropped from the matched data set. Optimal matching functions in a different manner; it is not linear, but allows for rearrangement of matches to make the overall sum of match distances as small as possible. Variable (or "full") matching uses an optimal matching algorithm, allowing a differing number of controls to be matched to each case and ensuring that: a control be used only once, at least one control gets matched to every case, and that all controls ultimately get matched to a case.

Our group wanted to match 3 controls to every case, but allow for a case to be matched to fewer controls (1 or 2) when 3 controls matched to the case on anywhere from 1 to 8 digits beyond the decimal point in the propensity score did not exist. Though the optimal matching algorithm allows for such varying number of control matches per case, our group lacked the resources to buy the needed SAS module (SAS OR) to perform this task. Relegated to already existing resources (e.g. SAS STAT module), we searched for methods of creating 1:N treated:untreated matches where N was allowed to vary across matched groups within the data set.

A very helpful SAS macro, %ONETOMANYMTCH, for performing 1:N case-control matches on propensity score existed in the literature (Parsons, 2004, SUGI 29). Particularly appealing was this macro's ability to create propensity

score matches across differing calipers (differences in propensity scores between cases and controls) within a matched group. Lacking, however, was the capability to select 1:N matches where N was allowed to vary across matched groups allowing, for example, some treated:untreated matches to be 1:3, others 1:2, and yet others 1:1.

This paper presents a new macro %MATCH1TO3 which, building on a very slightly modified version of the %ONETOMANYMTCH macro, provides a method for matching a variable number (from 1 to 3) of treateds:untreateds *within the same data set.* The benefit of this addition is the ability to retain treated patients/matched groups in a final analysis data set, even if they have fewer than the ideal number of untreated matches.

STEPS FOR CREATING 1:N TREATED:UNTREATED MATCHES (WHERE N CAN RANGE FROM 1 TO 3 ACROSS MATCHED GROUPS)

*For the entirety of this paper, the words **case** and **treated** will be used interchangeably, as will the words **untreated** and **control**.*

A. CREATE DATA SET CONTAINING, AT A MINIMUM, THE FOLLOWING VARIABLES (SEE APPENDIX A, SECTION A):

1. id

2. mycase (1=case, 0=control) - variable identifying case/treated status

3. pscore - propensity score, or other continuous variable on which cases and controls will be matched (BIG NOTE – BE SURE TO NAME THIS VARIABLE *pscore* to be compatible with programming in macro(s); can be changed, but will need to find *pscore* in the SAS code in Appendices A and B and replace it with the desired variable name

B. CREATE THE (SLIGHTLY MODIFIED) %ONETOMANYMTCH MACRO (SEE APPENDIX B)

Copy the SAS code from Appendix B into a SAS file, “including” the macro by incorporating the following code into main SAS program (See Appendix A, Section B):

```
%include "filepath\op_parsons_1toN_matching_macro.sas";
```

C. CREATE %MATCH1TO3 MACRO (SEE APPENDIX A, SECTION C)

This is a 1:N treated:untreated match macro (where N is allowed to range from 1 to 3 across matched groups) This new macro inputs the data set created in Step A, calls the ***%ONETOMANYMTCH*** from Step B, and creates the first treated:untreated (1:1) match. It then outputs a data set of controls matched in this first round and removes these controls from the pool of potential control matches for Round 2. The Round 2 pool of cases and potential controls is then used as input in a second call of ***%ONETOMANYMTCH***. A final (third) round of matching is performed, preparing the pool of potential matches for Round 3 in the same manner as described for Round 2.

D. OUTPUT FINAL MATCHED DATA SET WITH RATIOS OF TREATED:UNTREATEDS OF 1:1, 1:2, AND 1:3 (SEE APPENDIX A, SECTION D).

Below is a sample of records from a matched data set created by running the %MATCH1TO3 macro. Matched groups range in treated:untreated size from 1:1 to 1:3, determined by the availability of appropriate controls for each case. The bounds of “appropriate” are set within the%ONETOMANYMTCH macro, with controls matched to cases on anywhere from 8 digits (0.00000001) to 1 digit (0.1) past the decimal point. The most precise match (8 digit) is chosen if it exists; but if not, the macro cycles through potential matches at each decreasingly precise level until it finds a matched control or finally stops the process of looking for one when all remaining controls fail to match the case at the lowest acceptable level of precision (0.1) past the decimal point.

The strata variable in the below output identifies the group to which the case and matched control(s) belong. For example, strata 3 represents a matched group of one case (id=3) and two controls (ids 280 and 989) in which the propensity scores (pscore) range from ~0.17 to 0.22. The strata variable is the id of the case to which a given control “belongs”.

Obs	strata	mycase	id	pscore
1	3	1	3	0.02266227
2	3	0	280	0.01977699
3	3	0	989	0.01732024
4	5	1	5	0.48803240
5	5	0	292	0.48806750
6	5	0	729	0.48797331
7	5	0	891	0.48790652
8	6	1	6	0.15634250
9	6	0	508	0.15912056

E. COMPARE PROPENSITY SCORE MEANS OF ORIGINAL VS. MATCHED DATA SETS (SEE APPENDIX A, SECTION E)

Running the %MATCH1TO3 macro on a simulated data set of 100 cases (mycase=1) and 900 controls (mycase=0) resulted in the creation of 75 matched groups ranging in size from 1:1 to 1:3 matched cases:controls. The number of propensity score matched digits after the decimal point ranged from 8 to 1. Prior to matching, the difference in mean propensity score between cases and controls was 0.45; after matching this difference was reduced to~ 0.07.

Propensity Score Means prior to Matching

mycase	*Obs*	*N*	*Mean*	*Std Dev*	*Minimum*	*Maximum*
0	900	900	0.7455252	0.2889012	0.0081037	1.0000000
1	100	100	0.3063854	0.3100063	1.6443426E-6	0.9978912

Propensity Score Means after to Matching

mycase	*Obs*	*N*	*Mean*	*Std Dev*	*Minimum*	*Maximum*
0	178	178	0.4583173	0.2969706	0.0081037	0.9979417
1	75	75	0.3939231	0.3046613	0.0060743	0.9978912

Number of Matched Group Members per Strata

After using the %MATCH1TO3 macro described in this paper, there were 75 matched groups (strata) in the final data set (forty-five 1:3; thirteen 1:2; and seventeen 1:1 case:control matches). The 75 cases were matched on propensity score to their controls on anywhere from the first digit after the decimal place through 8 digits after the decimal place. Using the original %ONETOMANY macro would have resulted in the inclusion of 45% of the n=100 cases, all with a 1:3 case:control ratio rather than including 75% of the n=100 cases.

Strata Size	*# of Stratas of this Size*	*Percent*	*Cumulative # of Strata*	*Cumulative Percent*
2	17	22.67	17	22.67
3	13	17.33	30	40.00
4	45	60.00	75	100.00

CONCLUSION

This paper describes a macro (%MATCH1TO3) that, building on an already existing 1:N macro (%ONETOMANYMTCH), enables researchers to create matched case:control groups that vary in size from 1:1 to 1:3. This feature is particularly helpful in studies in which retaining the maximum number of cases in analysis data sets is important, even if the ideal number of matched controls is not available for all cases.

REFERENCES

Faries, D., Leon A., Maria Haro, J. and Obenchain, R. 2010. Analysis of Observational Health Care Data Using SAS, p. 24. Cary, NC: SAS Institute, Inc.

NCSS Statistical Software. “Data matching – Optimal and Greedy”. NCSS Statistical Software. Available at http://www.ncss.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Data_Matching-Optimal_and_Greedy.pdf

Parsons,Lori. Performing a 1:N Case-Control Match on Propensity Score. SUGI 29 Proceedings. Available at http://www2.sas.com/proceedings/sugi29/165-29.pdf .

Appendix A. Match1to3 Macro

```
/*******************************************************************************
*Date: 3/24/16
*Author: Janet Grubber/Carl Pieper
*Location:  Durham Veterans Affairs Medical Center, Health Services Research;
            Duke University Medical Center, Dept. of Biostatistics and Bioinformatics
*E-mail: janet.grubber@va.gov
*Program: pharmasug12016_p014_matching.sas  (PharmaSUG 2016 Poster PO14)
*Last modified:
*Purpose:  Creating Screened to Unscreened Matches in variable 1:3 ratio across
           Matched groups depending on availability of matches

*******************************************************************************/

options nofmterr mergenoby=error merror mprint;
libname matches "P:\OP_In_Men\Statistician\sas programs\pharmsug2016\output";


*creating sample data sets for use by macro;


*************************SECTION A - CREATE CASE-CONTROL DATA
SET(S)************************************;
data cases;
 input id pscore;
 cards;
 1    .3
 2    .4
 3    .5
 4    .51
 5    .6
 6    .7
 7    .
 ;
 run;


data controls;
 input id pscore;
 cards;
11    .29999999
12    .3
13    .31
21    .4
22    .45
31    .49999999
32    .5
33    .5
41    .50500001
42    .50000001
43    .51
51    .6
52    .60000002
53    .60000001
54    .6
55    .
56    .
 ;
run;
```

```
data casecont1;
  set cases controls;
  if 1 le id le 7 then mycase=1;
    else mycase=0;
run;



data casecont2;
 do id=1 to 100;
    pscore=ranuni(666)**3;
        mycase=1;
    output;
 end;
 do id=101 to 1000;
        pscore=1-ranuni(667)**3;
        mycase=0;
        output;
 end;
run;

**************************SECTION B - PULL IN SLIGHTLY TWEAKED VERSION OF ORIGINAL PARSONS
SUGI29 1:N MATCHING MACRO**********;

*pulling in slightly modified verion of Parsons original 1:N match macro (CODE IS PROVIDED IN
APPENDIX B);
%include "P:\OP_In_Men\Statistician\sas
programs\pharmsug2016\op_parsons_1toN_matching_macro.sas";

/*****************************************************************************************/
/*****************************************************************************************/
/* Original 1:N Matching Macro from Parsons SUGI29 */
/*****************************************************************************************/
/*****************************************************************************************/
/***********************************************/

/*%MACRO OneToManyMTCH (
      Lib,                      Library Name

      Dataset,                  Data set of all patients

      depend,           Dependent variable that indicates Case or Control
                              Code 1 for Cases, 0 for Controls
      PatientN,         Patient ID

      matches,                  Output data set of matched pairs
      NoContrls     Number of controls to match to each case;
*/

**************************SECTION C - CREATE NEW MACRO WHICH CALLS THE PARSONS 1:N
MACRO***************;

%macro match1to3 (dataset);

************************** Create Round 1 Matches (1:1 Match);


data matches.round1possmatches (keep=id mycase pscore);
  set &dataset (where=(pscore ne .));
 run;

*options nomprint;

*invoking Parsons SUGI29 macro;
%onetomanymtch(
Lib=matches,                /* Library Name */
Dataset=round1possmatches, /* Data set of all */
depend=mycase,              /* Dependent variable that indicates Case or Control - 1=cases,
0=controls */
patientn=id,
```

```
matches=mymatches1,         /* Output file of matched */
nocontrls=1);               /* pairs */


*options mprint;

*matched cases and 1st control;
data matches.round1matches;
  set matches.mymatches1;
  round=1;
  if mycase=1 then caseid=id;
    else if mycase=0 then contid=id;
run;

*saving matched controls from round1;
data matches.controls;
  set matches.round1matches(where=(mycase=0));
run;


************************** Round 2 Matches - controls identified  matches in this round will
increase the treated:untreated ratio to 1:2;

*creating pool of cases and controls for round 2 - removing matched controls from round 1;

proc sort data=matches.round1possmatches; by id; run;
proc sort data=matches.controls; by id; run;

data matches.round2possmatches;
  merge matches.round1possmatches (in=a)
        matches.controls (in=b);
               by id;
               if a=1 and b ne 1;
run;

*performing 2nd match - invoking Parsons SUGI29 macro a second time;

options nomprint;
%onetomanymtch(
Lib=matches,                /* Library Name */
Dataset=round2possmatches, /* Data set of all */
depend=mycase,              /* Dependent variable that indicates Case or Control - 1=cases,
0=controls */
patientn=id,
matches=mymatches2,         /* Output file of matched */
nocontrls=1);               /* pairs */

*options mprint;

*matched cases and 2nd control for specified year;
data matches.round2matches;
  set matches.mymatches2;
  round=2;
    if mycase=1 then caseid=id;
      else if mycase=0 then contid=id;
*run;
run;

*appending matched controls from round2 to cumulative controls file;
proc append base=matches.controls data=matches.round2matches (where=(mycase=0));
run;


************************** Round 3 Matches - controls identified  matches in this round will
increase the treated:untreated ratio to 1:2;

*creating pool of cases and controls for round 3 - removing matched controls from rounds 1 and 2;

proc sort data=matches.round2possmatches; by id; run;
proc sort data=matches.controls; by id; run;
```

```
data matches.round3possmatches;
  merge matches.round2possmatches (in=a)
        matches.controls (in=b);
               by id;
               if a=1 and b ne 1;
run;


*performing 3rd match - invoking Parsons SUGI29 macro a third time;

*options nomprint;

%onetomanymtch(
Lib=matches,                /* Library Name */
Dataset=round3possmatches, /* Data set of all */
depend=mycase,              /* Dependent variable that indicates Case or Control - 1=cases,
0=controls */
patientn=id,
matches=mymatches3,         /* Output file of matched */
nocontrls=1);               /* pairs */

*options mprint;

*matched cases and 3rd control;
data matches.round3matches;
  set matches.mymatches3;
  round=3;
    if mycase=1 then caseid=id;
      else if mycase=0 then contid=id;
run;


*appending matched controls from round2 to cumulative controls file;
proc append base=matches.controls
      data=matches.round3matches (where=(mycase=0));
run;


data tempmatches;
  set matches.round1matches
      matches.round2matches
      matches.round3matches;

         tempstrata=compress(round||"_"||match_1);
run;


proc sort data=tempmatches; by tempstrata descending mycase; run;

data tempmatches;
  set tempmatches; by tempstrata descending mycase;
  if first.tempstrata and mycase=1 then strata=id*1;
    else;
        retain strata;
  if mycase=1 and round in (2,3) then delete;
run;

**************************SECTION D - OUTPUT THE MATCHED DATA
SET*******************************************;

*outputting final matched data set with strata of matched cases and controls defined;
*removing duplicate cases;
proc sort data=tempmatches nodupkey
       out=matches.finalmatches (drop=caseid contid tempstrata match_1);
       by strata caseid contid; run;

proc sort data=matches.finalmatches; by strata descending mycase id; run;

title "Sample Final Matches, Data Set:  &dataset";
  proc print data=matches.finalmatches;
  var strata mycase id pscore;
```

```
  format pscore 11.8;
run;

**************************SECTION E - COMPARE PROPENSITY SCORE MEANS IN ORIGINAL VS. MATCHED
DATA SETS***********;
options ls=100;
title "Mean Propensity Score, Original Data Set before Match:  &dataset";
  proc means data=&dataset;
    class mycase;
       var pscore;
run;
title "Mean Propensity Score, after Match";
  proc means data=matches.finalmatches;
    class mycase;
       var pscore;
run;

title "Number Participants per Match";
  proc freq data=matches.finalmatches noprint;
  tables strata/out=checkstrata;
run;

proc freq data=checkstrata (rename=(count=numperstrata));
  tables numperstrata;
run;

title;
%mend;

*running macro;
%match1to3 (casecont1);
%match1to3 (casecont2);
```

Appendix B. Lori S. Parsons SUGI 29 Paper 165-29, *Performing a 1:N Case-Control Match on Propensity Score* (Slightly Modified by Grubber/Pieper)

```
/*****************************************************************************************/
/*****************************************************************************************/
/* Matching Macro */
/*****************************************************************************************/
/*****************************************************************************************/
%MACRO OneToManyMTCH (
      Lib,                        /* Library Name                                          */
      Dataset,           /* Data set of all patients
         */
      depend,            /* Dependent variable that indicates Case or Control       */
                         /* Code 1 for Cases, 0 for Controls                      */
      PatientN,  /* Patient ID
         */
      matches,           /* Output data set of matched pairs                      */
      NoContrls);        /* Number of controls to match to each case                */

               /*************************************************************************************/
               /* Macro to Create the Case and Control Data sets */
                         *Rounds the propensity score/probability to a certain number of digits;
                         *Initializes a match variable;
                         *Generates a random number for the controls;
                         *Separates out to two data sets;
               /*************************************************************************************/

               %MACRO INITCC(Dataset, /*CaseAndCtrlsls*/ /*Case and Controls in one dataset*/
                             digits        /*Number of digits to round the propensity score to */
                             );
            *cases;
                     data tcases (drop=cprob)
                           tctrl (drop=aprob) ;
                           set  &dataset /*&CaseAndCtrls.*/ ;

                           /* Create the data set of Controls */
                           if &depend. = 0 and pscore ne . then
                           do;
                                 cprob = Round(pscore,&digits.);
```

```
                  Cmatch = 0;
                  Length RandNum 8;
                  RandNum=ranuni(7654321);
                  Label RandNum='Uniform Randomization Score';
                  output tctrl;
            end;

            /* Create the data set of Cases */
            else if &depend. = 1 and pscore ne . then
            do;
                  Cmatch = 0;
                  aprob =Round(pscore,&digits.);
                  output tcases;
            end;
      run;
%SORTCC;
%MEND INITCC;

/*************************************************************************************************/
/* Macro to sort the Cases and Controls data set */
/*************************************************************************************************/
%MACRO SORTCC;
      *Sort the cases;
      proc sort data=tcases out=&LIB..Scase;
            by pscore;
      run;

          *sort the controls;
      proc sort data=tctrl out=&LIB..Scontrol;
            by pscore randnum;
      run;
%MEND SORTCC;



/*************************************************************************************************/
/* Macro to Perform the Match */
/*************************************************************************************************/
%MACRO MATCH (MATCHED,DIGITS);

      data &lib..&matched. (drop=Cmatch randnum aprob cprob start oldi curctrl matched);

            /* select the cases data set */
            set &lib..SCase ;

                        *Suppose curob refers to the current observation number;

            curob + 1;
            matchto = curob;
            if curob = 1 then do;
                  start = 1;
                  oldi = 1;
            end;


            /* select the controls data set */
            DO i = start to n;
                  set &lib..Scontrol point = i nobs = n; *creates a one to many pairing;
                                                         *pairs each case with all of the
                                                          possible controls;
                  if i gt n then goto startovr;
                  if _Error_ = 1 then abort;
                  curctrl = i;

                  /* output control if match found */
                  if aprob = cprob then
                  do;                                    *probabilities are equal;
                        Cmatch = 1;                      *Mark as matched;
                        output &lib..&matched.;          *Output to matched dataset;
                        matched = curctrl;               *record observation number of the
                                                          matched;
                        goto found;
                  end;

                  /* exit do loop if out of potential matches */
                  else if cprob gt aprob then
                        goto nextcase;

                  startovr: if i gt n then
                  goto nextcase;
            END; /* end of DO LOOP */
```

```
            /* If no match was found, put pointer back*/
            nextcase:
            if Cmatch=0 then start = oldi;

            /* If a match was found, output case and increment pointer */
            found:
            if Cmatch = 1 then do;
                  oldi = matched + 1;
                  start = matched + 1;
                        set &lib..SCase point = curob;
                        output &lib..&matched.;
            end;

            retain oldi start;
            if _Error_=1 then _Error_=0;
      run;

      /* get files of unmatched cases and controls */
      proc sort data=&lib..scase out=sumcase;
            by &PatientN.;
      run;

      proc sort data=&lib..scontrol out=sumcontrol;
            by &PatientN.;
      run;

      proc sort data=&lib..&matched. out=smatched (keep=&PatientN. matchto);
            by &PatientN.;
      run;

      data tcases (drop=matchto);
            merge sumcase(in=a) smatched;
            by &PatientN.;
            if a and matchto = . ;
            cmatch = 0;
            aprob =Round(pscore,&digits.);
      run;

      data tctrl (drop=matchto);
            merge sumcontrol(in=a) smatched;
            by &PatientN.;
            if a and matchto = . ;
            cmatch = 0;
            cprob = Round(pscore,&digits.);
      run;
      %SORTCC

%MEND MATCH;

/*****************************************************************************************/
/* Macro to call Macro MATCH for each of the 8-digit to 1-digit matchs */
/*****************************************************************************************/
%MACRO CallMATCH;
      /* Do a 8-digit match */
      %MATCH(Match8,.00000001);
      /* Do a 7-digit match on remaining unmatched*/
      %MATCH(Match7,.0000001);
      /* Do a 6-digit match on remaining unmatched*/
      %MATCH(Match6,.000001);
      /* Do a 5-digit match on remaining unmatched*/
      %MATCH(Match5,.00001);
      /* Do a 4-digit match on remaining unmatched */
      %MATCH(Match4,.0001);
      /* Do a 3-digit match on remaining unmatched */
      %MATCH(Match3,.001);
      /* Do a 2-digit match on remaining unmatched */
      %MATCH(Match2,.01);
      /* Do a 1-digit match on remaining unmatched */
      %MATCH(Match1,.1);
%MEND CallMATCH;

/*****************************************************************************************/
/* Macro to combine/set all the matches files into one file */
/*****************************************************************************************/
%MACRO MergeFiles(MatchNo);
      data &matches.&MatchNo. (drop = matchto);
            set &lib..match8(in=a) &lib..match7(in=b) &lib..match6(in=c) &lib..match5(in=d)
                  &lib..match4(in=e)
                  &lib..match3(in=f) &lib..match2(in=g) &lib..match1(in=h);
            if a then match_&MatchNo. = matchto;
```

```
                  if b then match_&MatchNo. = matchto + 10000;
                  if c then match_&MatchNo. = matchto + 100000;
                  if d then match_&MatchNo. = matchto + 1000000;
                  if e then match_&MatchNo. = matchto + 10000000;
                  if f then match_&MatchNo. = matchto + 100000000;
                  if g then match_&MatchNo. = matchto + 1000000000;
                  if h then match_&MatchNo. = matchto + 10000000000;
            run;
%MEND MergeFiles;

/*****************************************************************************************/
/*****************************************************************************************/
/* Perform the initial 1:1 Match */
/*****************************************************************************************/
/*****************************************************************************************/

/* Create file of cases and controls */
%INITCC(&LIB..&dataset.,.00000001);

/* Perform the 8-digit to 1-digit matches */
%CallMATCH;

/* Merge all the matches files into one file */
%MergeFiles(1)

/*****************************************************************************************/
/*****************************************************************************************/
/* Perform the remaining 1:N Matches */
/*****************************************************************************************/
/*****************************************************************************************/
%IF &NoContrls. gt 1 %Then %DO;
      %DO i = 2 %TO &NoContrls.;

            %let Lasti=%eval(&i. - 1);

         /* ********** */
         /* Start with Cases from the last Matched Cases file and the remaining Un-Matched */
         /* Controls. NOTE: The Unmatched Controls file (Scontrol) is created at end of the */
         /* previous match                                                                  */

            /* Select the Matched Cases from the last Matched File */
            data &LIB..Scase;
                  set &matches.&Lasti.;
                  where &Depend. = 1;
            run;

            /* ********** */
            /* Perform the 8-1 digit matches between Matched Cases and the Unmatched Controls
*/
            %CallMATCH;

            /* ********** */
            /* Merge the 8-digit to 1-digit matches files into one file */
            %MergeFiles(&i.)

            %DO m = 1 %TO &Lasti.;
                  data &matches.&i.;
                        set &matches.&i.;
                        if &Depend.=0 then Match_&m. = .;
                  run;
            %END;

            /* ********** */
            /* Determine which OLD Controls correspond to the kept Cases */

            %DO c = 1 %TO &Lasti.;
                  /* Select the KEPT Cases */
                  proc sort data=&matches.&i. out=skeepcases (keep = Match_&c.);
                        by Match_&c.;
                        where &Depend. = 1;
                  run;
                  /* Get the OLD Controls */
                  proc sort data = &matches.&Lasti. out = soldcontrols&c.;
                        by Match_&c.;
                        where &Depend. = 0 and Match_&c. ne . ;
                  run;
                  /* Get the OLD Controls that correspond to the kept Cases */
                  data keepcontrols&c.;
                        merge skeepcases (in = a) soldcontrols&c. (in = b);
                        by Match_&c.;
                        if a;
```

```
                              run;
                        %END;

                        /* ********** */
                        /* Combine all the OLD Controls into one file */

                        data keepcontrols;
                              set keepcontrols1 (obs=0);
                        run;
                        %DO k = 1 %TO &Lasti.;
                              data keepcontrols;
                                    set keepcontrols keepcontrols&k.;
                              run;
                        %END;

                        /* ********** */
                        /* Append the OLD matched Controls to the new file of matched cases and controls
*/
                        data &matches.&i.;
                              set &matches.&i. keepcontrols;
                        run;

                        /* ********** */
                        /* If there are more matches to be made, add the previously matched, but not kept,
*/
                        /* controls back into the pool of unmatched controls */
                        %if &i. lt &NoContrls. %then %do;

                              %DO z = 1 %TO &Lasti.;

                                    /* Select all the KEPT Cases */
                                    proc sort data=&matches.&i. out=skeepcases (keep = Match_&z.);
                                          by Match_&z.;
                                          where &Depend. = 1;
                                    run;

                                    /* Select all the OLD Controls */
                                    proc sort data = &matches.&Lasti. out = soldcontrols&z.;
                                          by Match_&z.;
                                          where &Depend. = 0 and Match_&z. ne .;
                                    run;

                                    /* Keep the OLD Controls that correspond to the NOT KEPT Cases */
                                    /* Drop the previous Match_X variable */
                                    data AddBackControls&z. (drop = Match_&z.);
                                          merge skeepcases (in = a) soldcontrols&z. (in = b);
                                          by Match_&z.;
                                          if b and not a;
                                    run;
                              %END; /* End DO */

                              /* Drop the previuos Match_X variable */
                              data &LIB..Scontrol (drop = Match_&lasti. );
                                    set &LIB..Scontrol;
                              run;
                              /* Append */
                              %DO y = 1 %TO &Lasti.;
                                    data &LIB..Scontrol;
                                          set &LIB..Scontrol AddBackControls&y.;
                                    run;
                              %END; /* End DO */
                        %end; /* End IF */
                  %END; /* End Main DO */
                  %END; /* End Main IF */

            /*************************************************************************************************/
            /*************************************************************************************************/
            /* Save the final matched pairs data set */
            /*************************************************************************************************/
            /*************************************************************************************************/

            /* Sort file by the dependent/Treatment/variable that indicates group Variable */
                  proc sort data=&matches.&NoContrls. out = &lib..&matches.;
                        by &depend.;
                  run;

%MEND OneToManyMTCH;

*****************************************************************************************************;
```