DOCUMENTS DE TRAVAIL CEMOI / CEMOI WORKING PAPERS

A SAS macro to estimate Average Treatment Effects with Propensity Score Matching

Nicolas Moreau[1]

http://cemoi.univ-reunion.fr/publications/

Centre d'Economie et de Management de l'Océan Indien
Université de La Réunion

October 2016

Abstract

This paper presents a SAS macro to estimate the Average Treatment Effect (ATE) and the Average Treatment Effect for the Treated (ATET) based on propensity score with nearest-neighbor matching. The robust standard errors derived in Abadie and Imbens (2016) are computed.

JEL : C210

[1] E-mail: nicolas.moreau@univ-reunion.fr

Introduction

In this paper we present the SAS macro *ps_matching*. *ps_matching* estimates the Average Treatment Effect (ATE) and the Average Treatment Effect for the Treated (ATET) based on propensity scores with nearest-neighbor matching with replacement. The adjusted standard errors derived in Abadie and Imbens (2016) are computed. They account for the fact that the propensity score is estimated in a first step. We refer the reader to this article for a clear and comprehensive presentation of the propensity score matching estimator.
The source code is included in Appendix to this paper and is also available from the author.

Syntax of *ps_matching*

The syntax is %ps_matching(data=,y=,w=,x=,M=,Link=,L=,Lt=);

Data specifies the data set. *y* is the outcome variable. *w* is the binary variable treatment indicator. *x* specifies the list of covariates to be used in the matching.
M specifies the number of matches to be made per observation. It could be any integer between 1 and the minimum of the number of treated and controls in the sample. If there are ties, if different matched pairs (*i,j*) and (*i,l*) lead to the same score distance $d_{ij}= d_{il}$, the number of matches per unit is greater than *M*.
Link=probit requests a probit estimation of the propensity score to be used instead of the default logit model when *Link* is not specified by the user.
L is a small positive integer (typically, *L*=2) needed to compute the variance estimator of ATE (see Abadie and Imbens, 2016). *Lt* is another small positive integer (typically, *Lt*=1) needed to compute the variance estimator of ATET (see again Abadie and Imbens, 2016).

Note that all variables in *y*, *x* and *w* must be numeric.

Results presentation and output data files

The first two tables summarize the model specification and the estimation options. The third table gives summary statistics, such as the number of treated units, the number of controls matched to treated units and so on. The last table shows the main results. The column "Estimate" reports the estimated average treatment effects. The next column shows their robust standard errors. "z" are the z-statistics used to test whether the average treatment effects are 0. They are computed as the estimated parameters divided by their standard error. The column "P-value" reports the p-values for the z-statistics for a two-sided test. The last two columns exhibit the lower and upper bounds of the 95% confidence interval for the z-statistics.
A box-plot is then provided to evaluate the validity of the common support assumption. To assess the validity of the balancing hypothesis and compare the covariate distributions between treatment groups before and after matching, normalized differences (see Austin, 2009), box-plots and empirical distribution functions are provided for five intervals of the estimated propensity score. These intervals are constructed from the 20th, 40th, 60th and 80th percentiles of the empirical distribution of the estimated propensity score. These different plots are supplied for continuous variables only. For binary variables, contingency tables are provided instead.
In *ps_matching*, all covariates that are not binary are considered continuous.

Two temporary output data files are created. They need to be stored in a specific folder with a libname statement to become permanent files.
Outdata1 includes an internal identification number for observation *i* created by the program based on the original sort order and called *_id_*. Il also includes the outcome variable *y*, the covariates *x* and the treatment group indicator *w*. *CardJMi* specifies the number of matches for unit *i*. *Count* is

the number of times unit *i* is used as a match. *Km_i* specifies the number of times unit *i* is used as a matched for any observation *j* of the opposite treatment group weighted by the total number of matches for the given observation *j*. *Pscore_pred* is the estimated propensity score for observation *i*.

Outdata2 includes the list of indices for the *M* closest matches for unit *i*. *_id_* is the internal identification number for observation *i* and *idM* the corresponding identification number of *i*'s closest matches in the opposite treatment group.
For each *_id_*, there is one row per match. For instance, if unit 3 is matched with units 5, 6 and 10, there are three rows in *outdata2* that correspond to *_id_*=3; the first one with *idM*=5, the second one with *idM*=6 and the last one with *idM*=10. For each matched pair (*i*,*j*), the score distance (in absolute value) between unit *i* and unit *j* of the opposite treatment group is stored, so as the unit *j*'s outcome value.

An example

We will illustrate the use of *ps_matching* by using data from a study of the effect of a mother's smoking status during pregnancy on infant birth weight as reported by Cattaneo (2010) and used in Stata Treatment-Effects Reference Manual, release 14.

bweight: infant birth weight
dummy_smoke= 1 if mother smoked during pregnancy, 0 otherwise
dummy_married= 1 if married, 0 otherwise
mage: mother's age
mage2: squared mother's age
medu: mother's education in years
dummy_fbaby=1 if mother's first birth, 0 otherwise.

The instructions %ps_matching(data=cattaneo2,y=bweight,w=dummy_smoke,x=dummy_married mage mage2 medu dummy_fbaby,M=1,Link=,L=2,Lt=1);

give the following results:

ESTIMATING AVERAGE TREATEMENT EFFECTS with PROPENSITY SCORE MATCHING

ModelSpecification

Outcome variable:	bweight
Binary treatment:	dummy_smoke
Matching variables:	dummy_married mage mage2 medu dummy_fbaby

EstimationOptions

Number of matches requested (M):	1
Number of matches requested in the same treatment group (L and Lt):	2 and 1

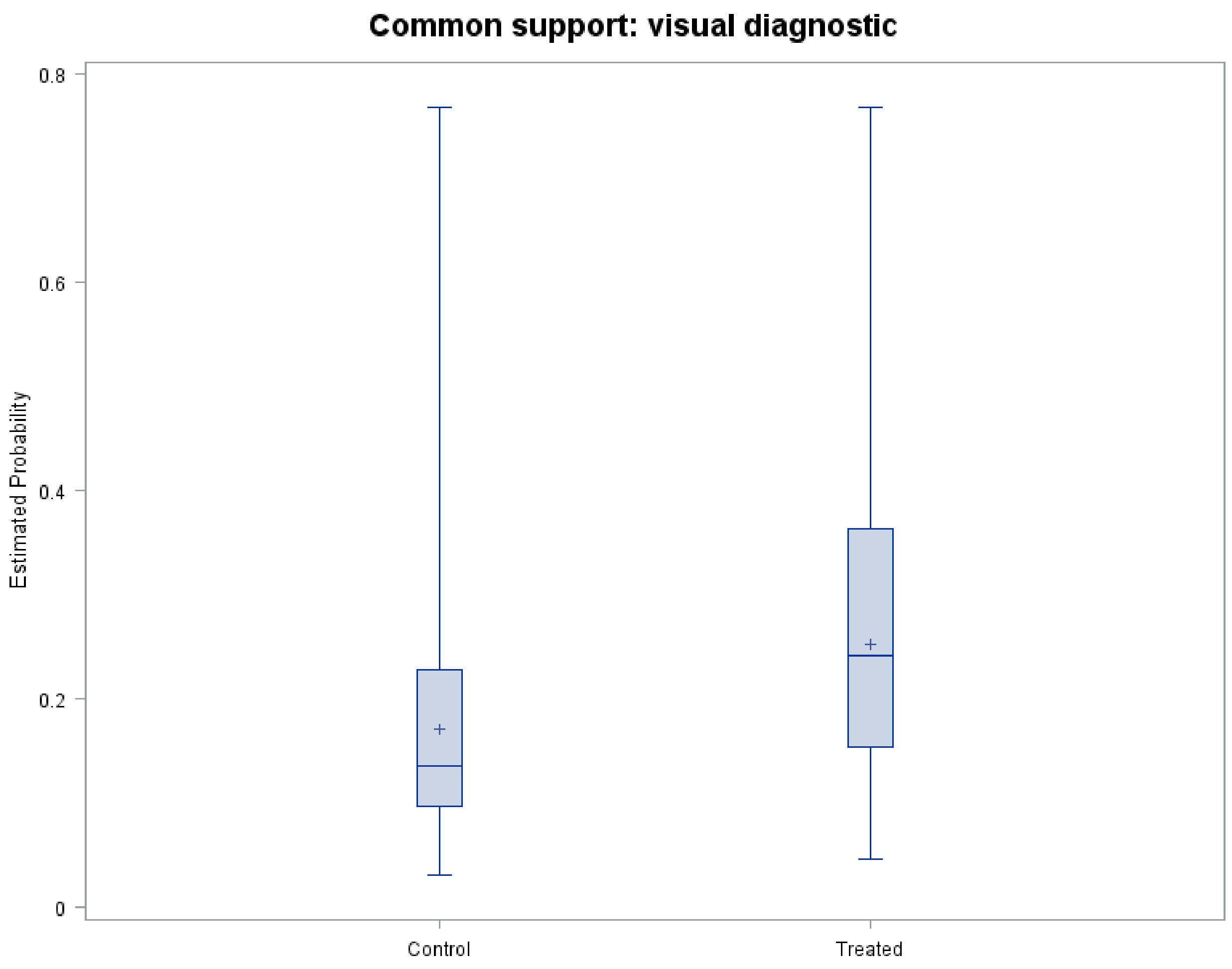
SummaryStatistics

Number of observations:	4642
Number of control units:	3778
Number of treated units:	864
Number of treated units matched to controls:	813
Number of control units matched to treated:	2804
Number of times a treated unit is used as a match (MIN):	1
Number of times a treated unit is used as a match (MAX):	75
Number of times a control unit is used as a match (MIN):	1
Number of times a control unit is used as a match (MAX):	19

Estimation results with Abadie and Imbens (2016) robust standard errors

Results

	Estimate	**Std.Error**	**z**	**P-value**	**L. bound 95% CI**	**U. bound 95% CI**
Average Treatment Effect (ATE)	-210.952	33.047	-6.383	0	-71.155	58.388
Average Treatment Effect for the Treated (ATET)	-237.391	26.625	-8.916	0	-61.101	43.269



Normalized covariate mean differences between treated and controls

	Unmatched Sample	Matched sample (T)	Matched sample (C)
DUMMY_MARRIED	-0.596	-0.514	-0.579
MAGE	-0.3	-0.136	-0.344
MAGE2	-0.303	-0.13	-0.348
MEDU	-0.547	-0.472	-0.518
DUMMY_FBABY	-0.166	-0.145	-0.154

Note: 'Matched sample (T)' is for normalized mean differences between all sample treated and their matches, 'Matched sample (C)' for normalized mean differences between all sample controls and their matches.

Comparing binary covariate distributions between treatment groups before and after matching
Interval 1 of the estimated propensity score

			SampleType			
			All treated	All controls	Matched controls	Matched treated
dummy_married						
	0	Distribution in %	.	0.12	.	.
	1	Distribution in %	100.00	99.88	100.00	100.00
dummy_fbaby						
	0	Distribution in %	26.09	30.81	26.09	23.69
	1	Distribution in %	73.91	69.19	73.91	76.31

Comparing binary covariate distributions between treatment groups before and after matching
Interval 2 of the estimated propensity score

			SampleType			
			All treated	All controls	Matched controls	Matched treated
dummy_married						
	0	Distribution in %	.	0.24	.	.
	1	Distribution in %	100.00	99.76	100.00	100.00
dummy_fbaby						
	0	Distribution in %	22.55	34.83	22.77	31.73
	1	Distribution in %	77.45	65.17	77.23	68.27

Comparing binary covariate distributions between treatment groups before and after matching
Interval 3 of the estimated propensity score

			SampleType			
			All treated	All controls	Matched controls	Matched treated
dummy_married						
	0	Distribution in %	3.33	2.27	2.82	1.11
	1	Distribution in %	96.67	97.73	97.18	98.89
dummy_fbaby						
	0	Distribution in %	86.67	94.53	89.44	96.35
	1	Distribution in %	13.33	5.47	10.56	3.65

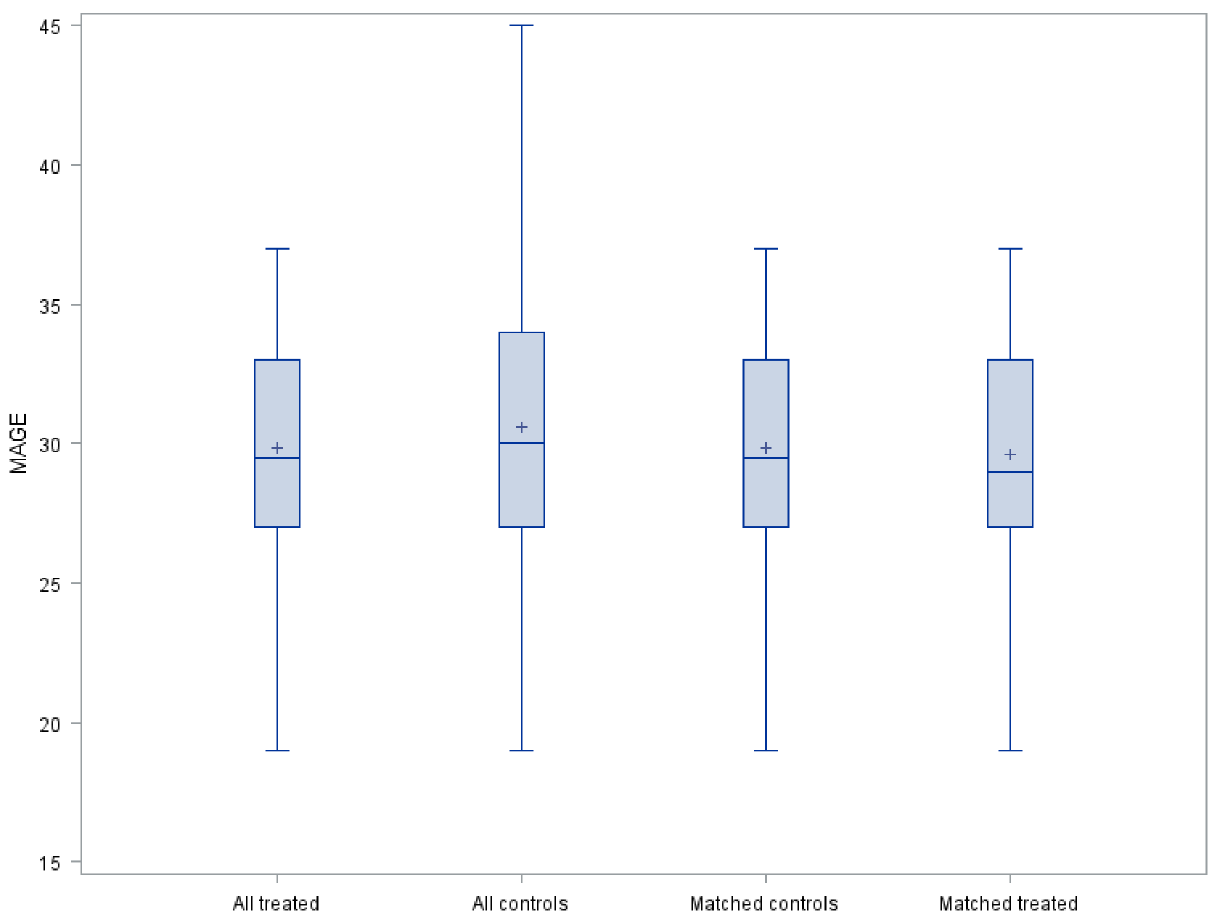
Comparing binary covariate distributions between treatment groups before and after matching
Interval 4 of the estimated propensity score

			SampleType			
			All treated	All controls	Matched controls	Matched treated
dummy_married						
	0	Distribution in %	50.44	47.89	51.39	48.20
	1	Distribution in %	49.56	52.11	48.61	51.80
dummy_fbaby						
	0	Distribution in %	49.56	53.48	48.61	50.65
	1	Distribution in %	50.44	46.52	51.39	49.35

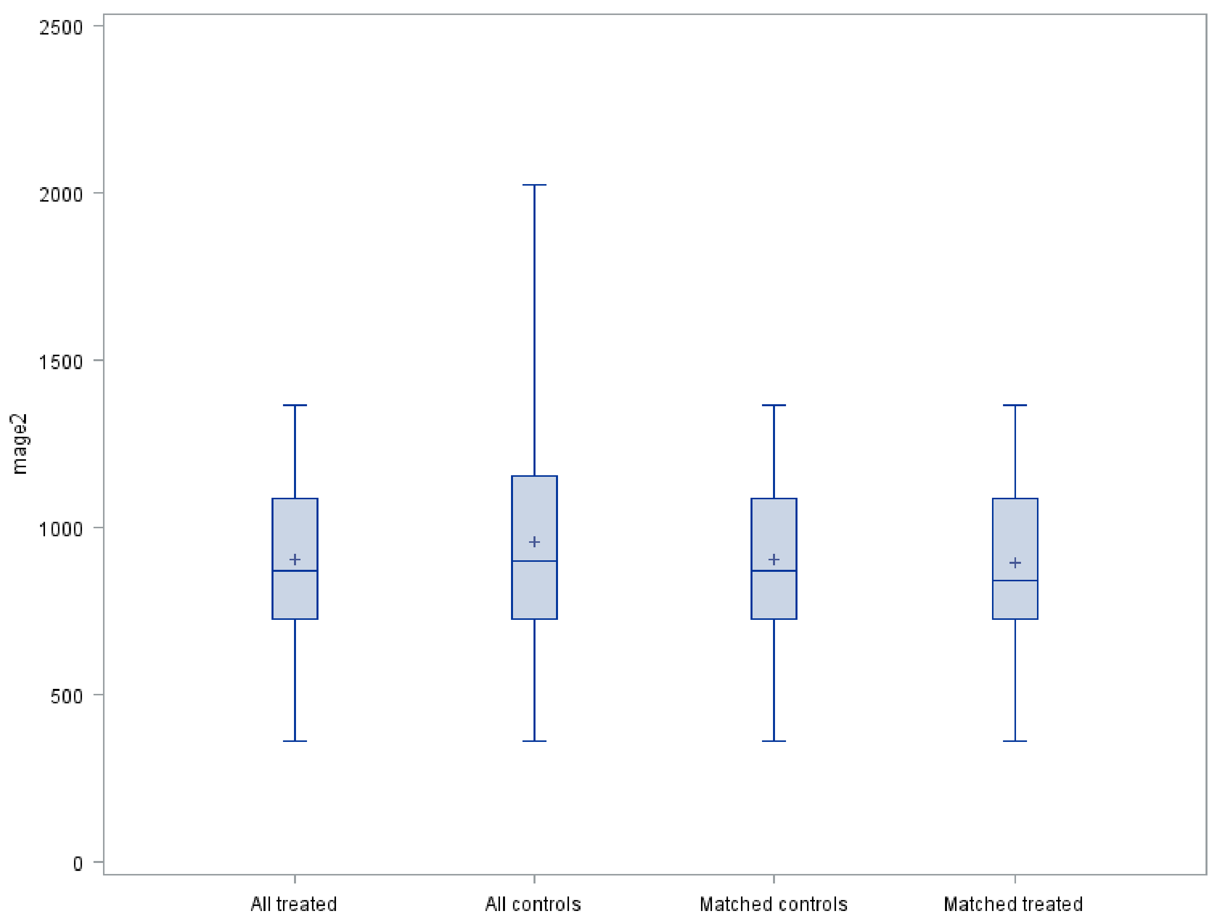
Comparing binary covariate distributions between treatment groups before and after matching
Interval 5 of the estimated propensity score

		SampleType			
		All treated	All controls	Matched controls	Matched treated
dummy_married					
0	Distribution in %	99.11	96.11	99.68	99.59
1	Distribution in %	0.89	3.89	0.32	0.41
dummy_fbaby					
0	Distribution in %	78.40	68.70	77.60	68.09
1	Distribution in %	21.60	31.30	22.40	31.91

Checking balance through box-plots, interval 1 of the estimated propensity score distribution

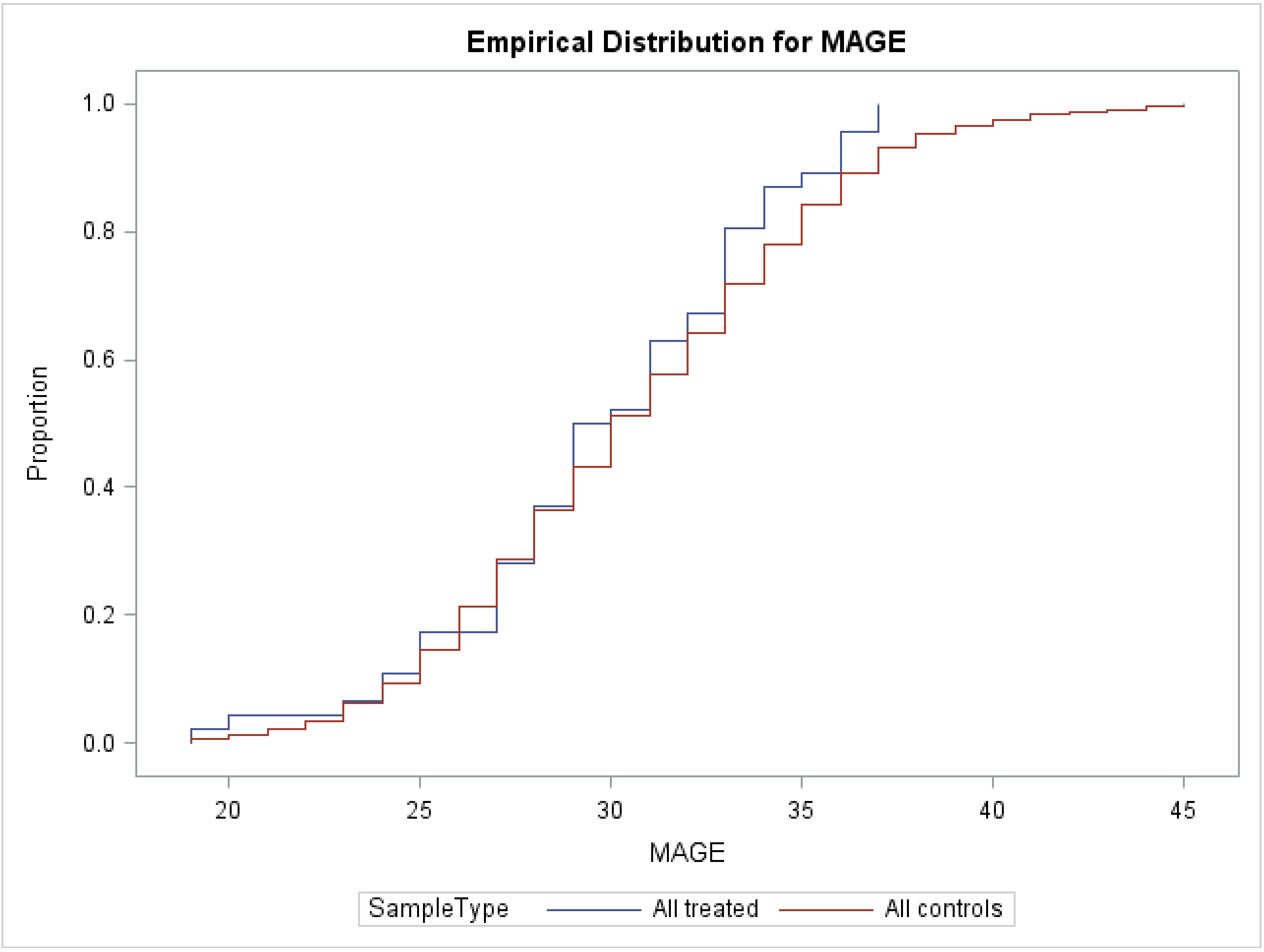


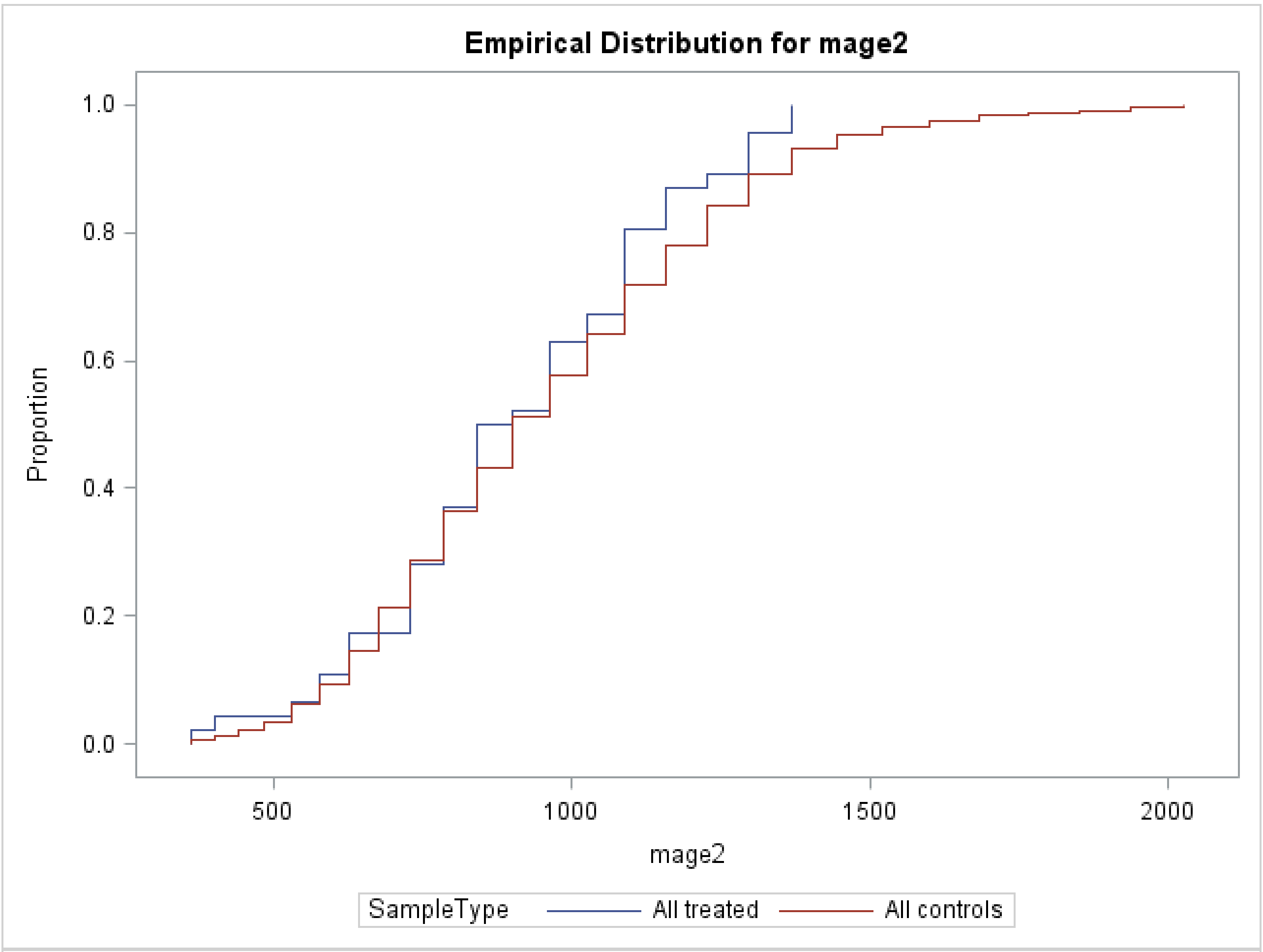
Checking balance through box-plots, interval 1 of the estimated propensity score distribution

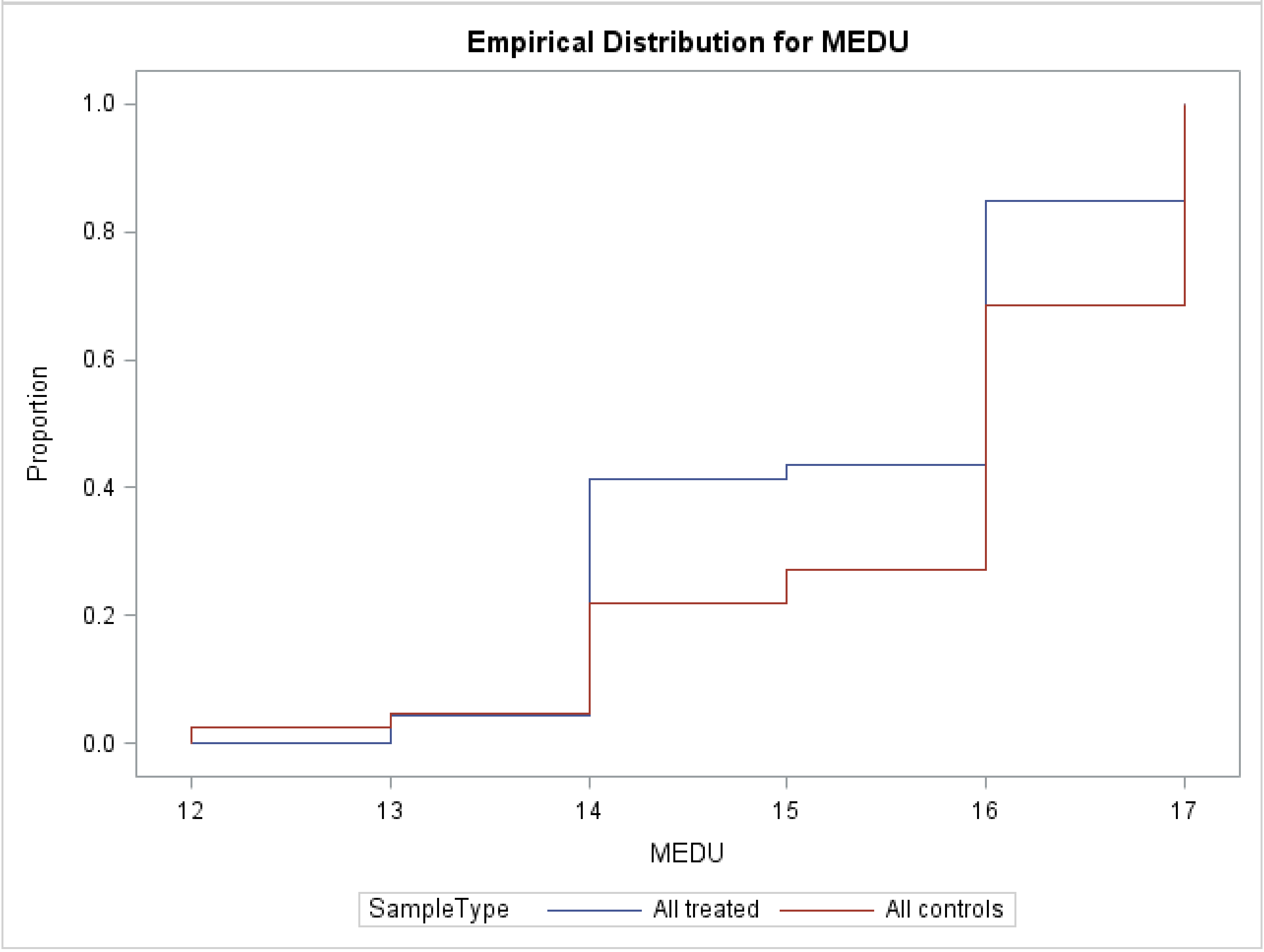


Checking balance through empirical distribution functions, interval 1 of the estimated propensity score distribution

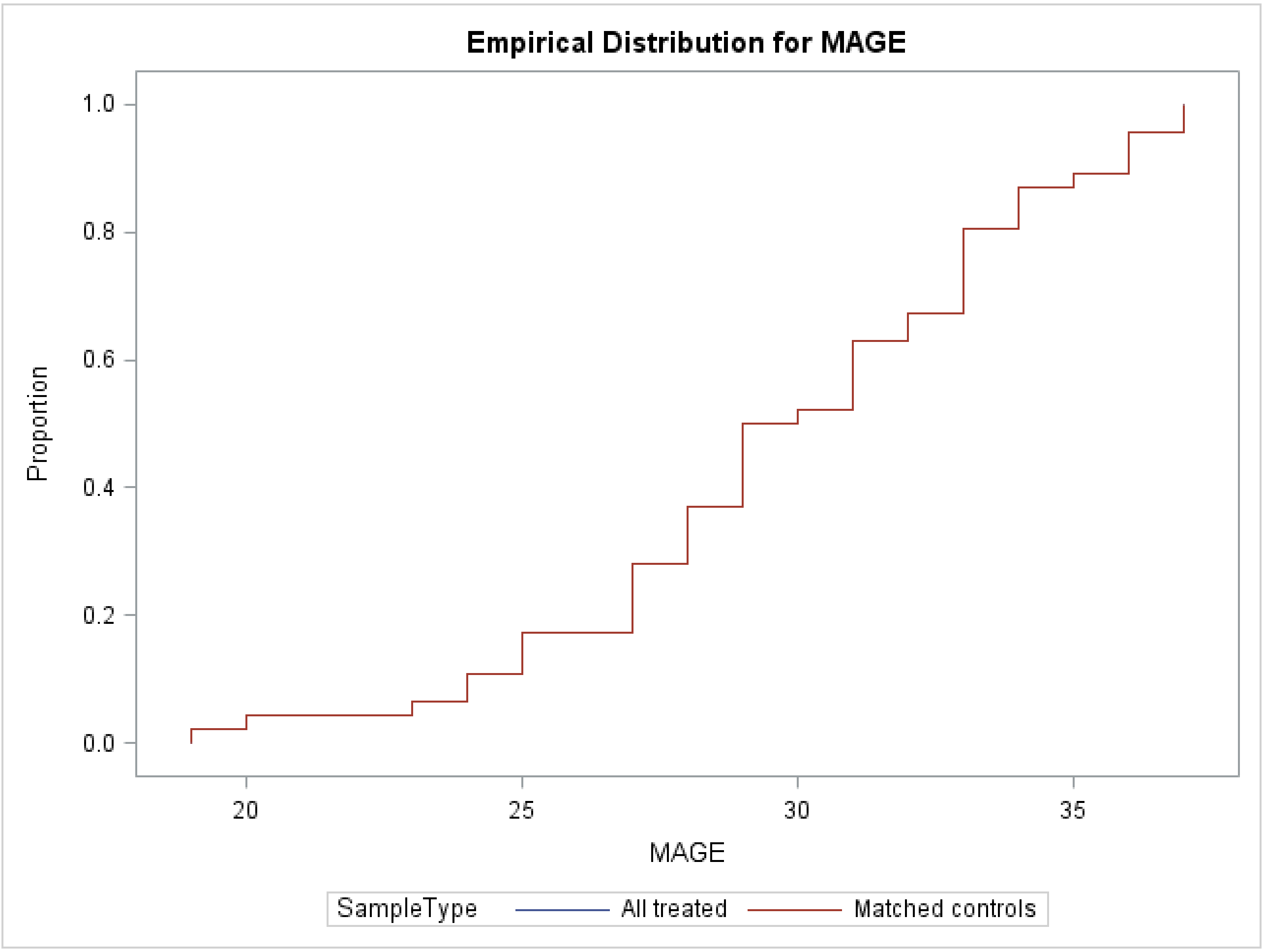
Original sample of treated vs original sample of controls

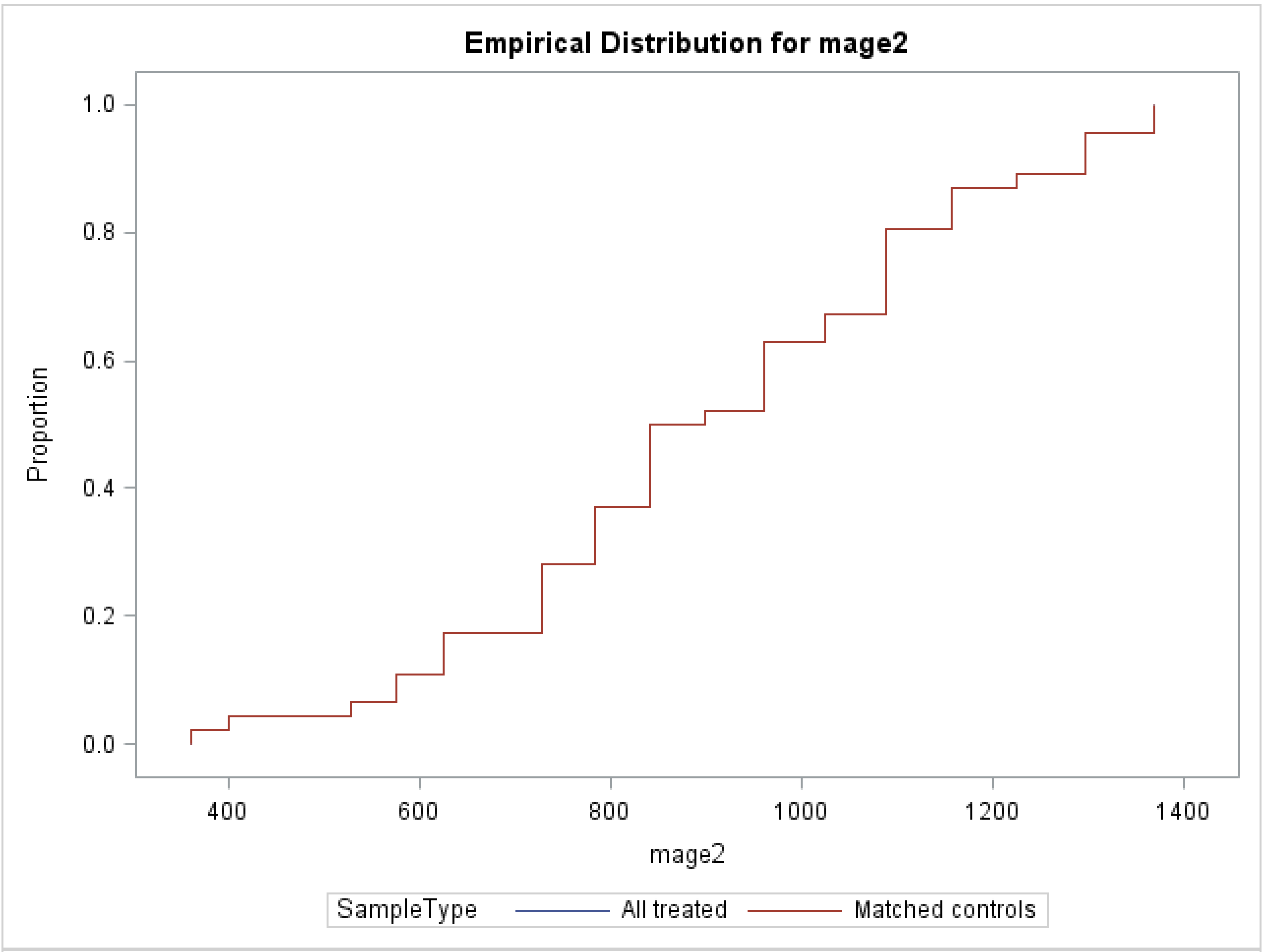


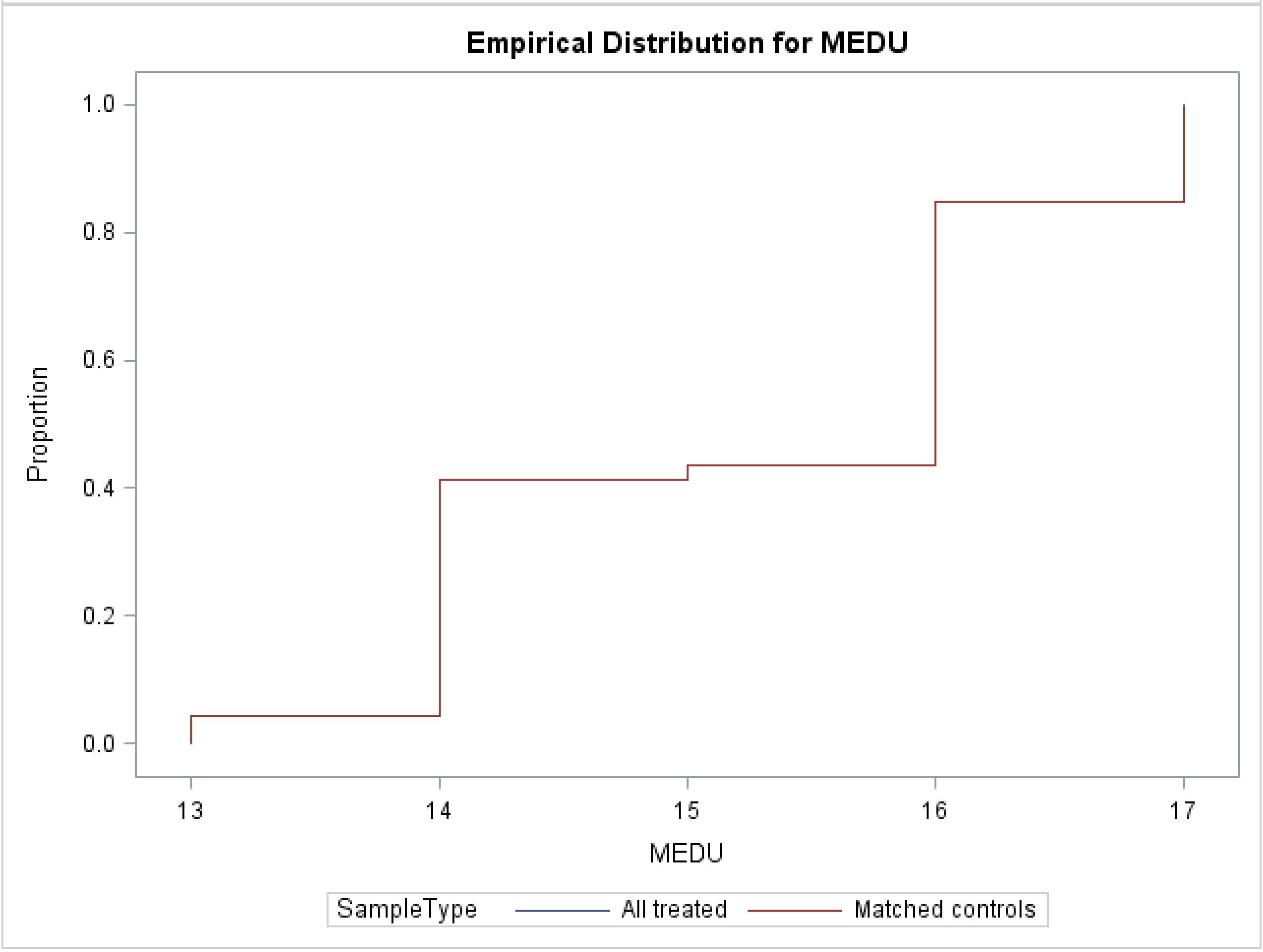

Empirical Distribution for mage2
Proportion
1.0
0.8
0.6
0.4
0.2
0.0
500
1000
1500
2000
mage2
SampleType
All treated
All controls


Empirical Distribution for MEDU
Proportion
1.0
0.8
0.6
0.4
0.2
0.0
12
13
14
15
16
17
MEDU
SampleType
All treated
All controls

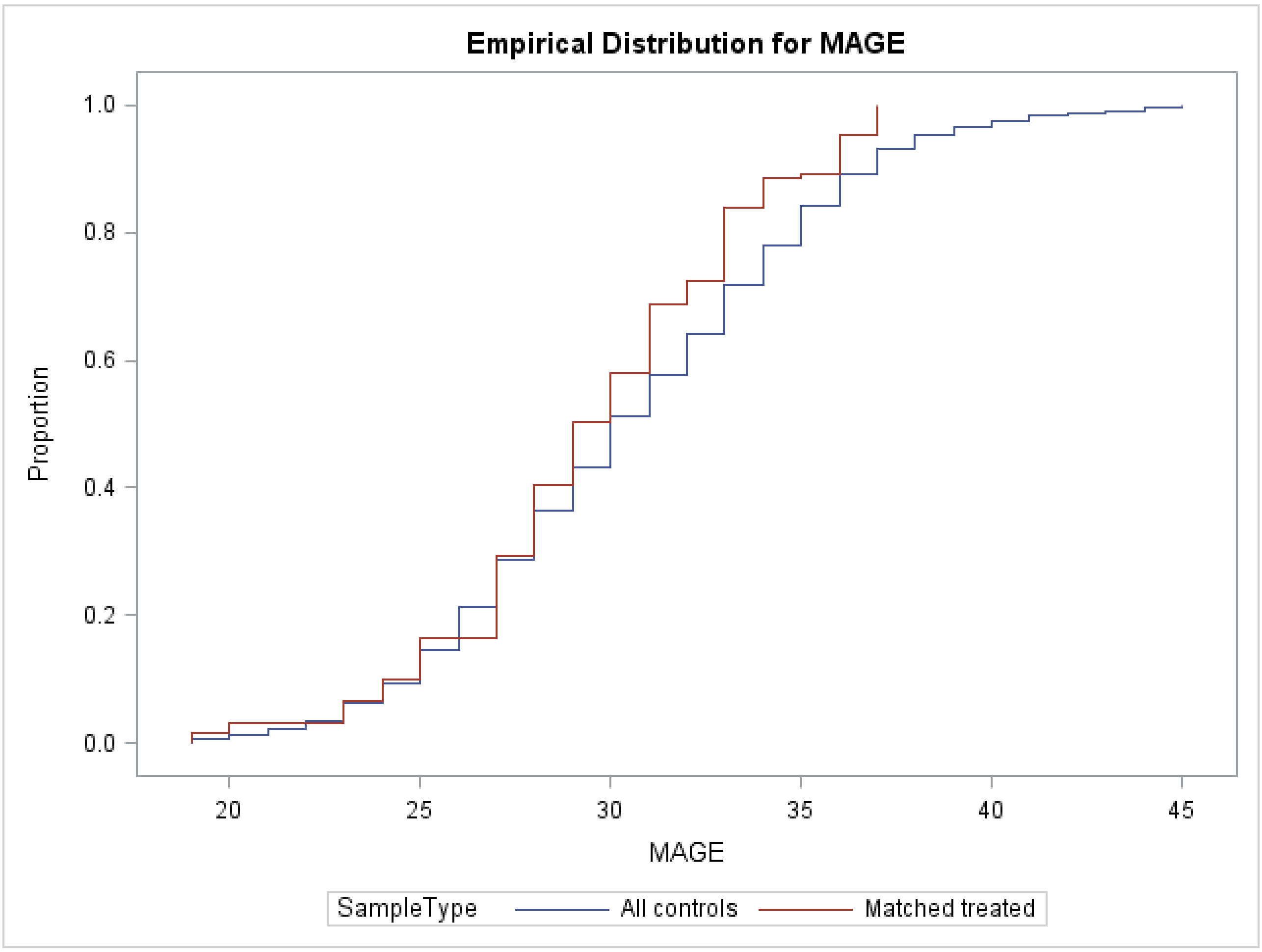
Checking balance through empirical distribution functions, interval 1 of the estimated propensity score distribution
Original sample of treated vs final sample of matched controls

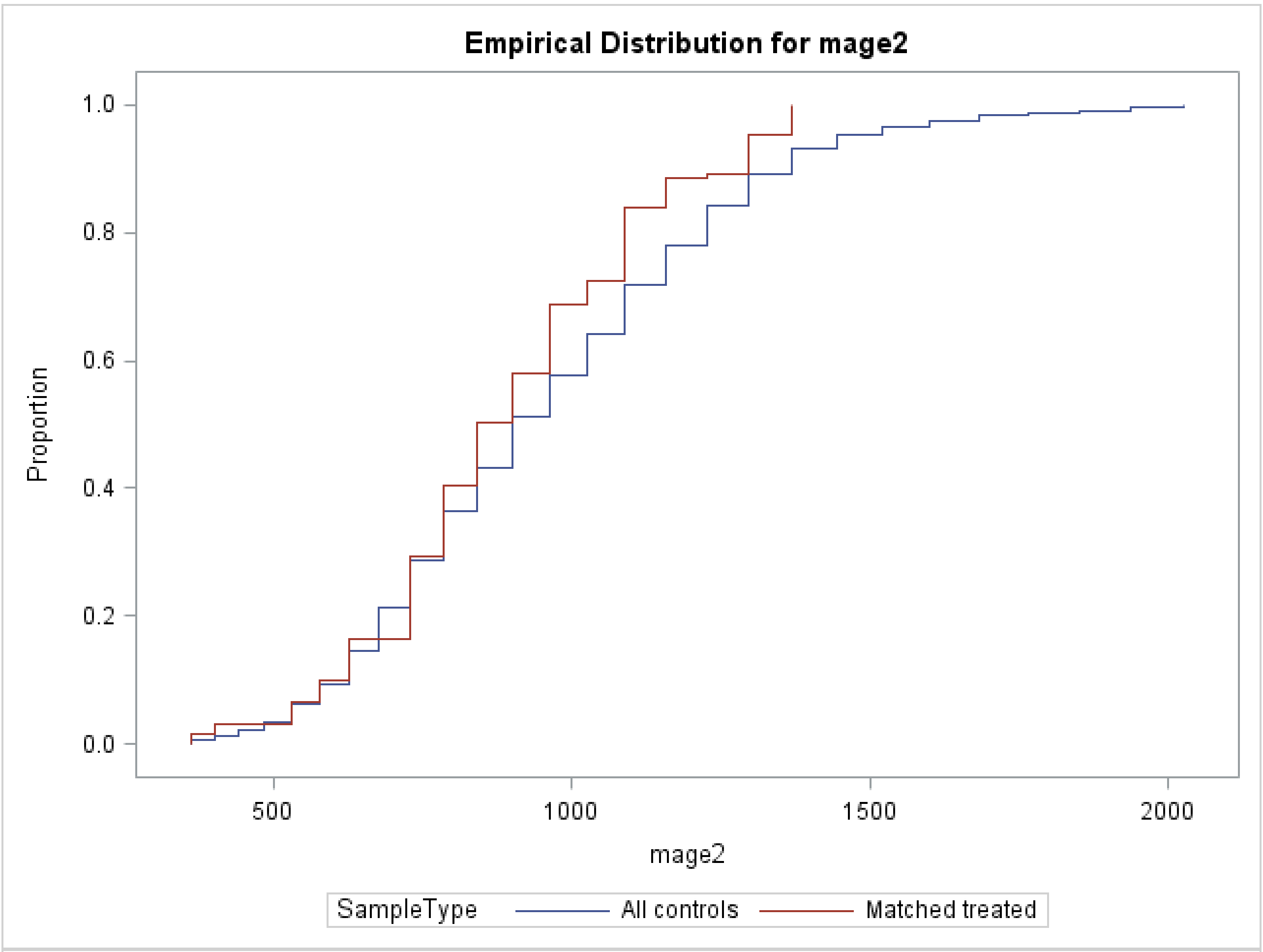


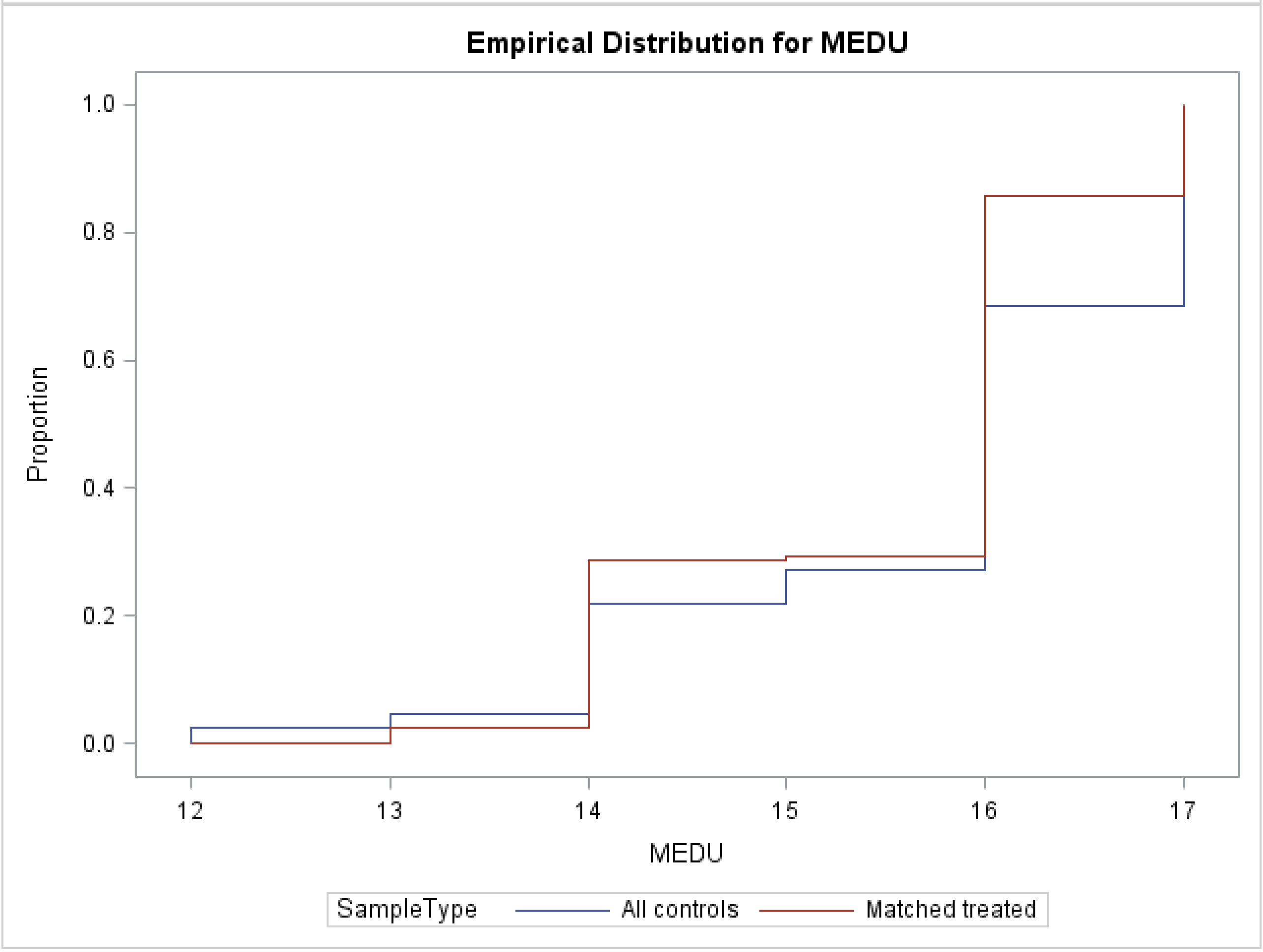

Empirical Distribution for mage2
1.0
0.8
0.6
0.4
0.2
0.0
Proportion
400
600
800
1000
1200
1400
mage2
SampleType
All treated
Matched controls


Empirical Distribution for MEDU
1.0
0.8
0.6
0.4
0.2
0.0
Proportion
13
14
15
16
17
MEDU
SampleType
All treated
Matched controls

Checking balance through empirical distribution functions, 1th interval of the estimated propensity score distribution
Original sample of controls vs final sample of matched treated




Empirical Distribution for mage2
Proportion
1.0
0.8
0.6
0.4
0.2
0.0
500
1000
1500
2000
mage2
SampleType
All controls
Matched treated


Empirical Distribution for MEDU
Proportion
1.0
0.8
0.6
0.4
0.2
0.0
12
13
14
15
16
17
MEDU
SampleType
All controls
Matched treated

and so on for intervals 2 to 5 of the estimated propensity score distribution.

References

Abadie A., Drukker D., Leber Herr J. and Imbens G. W. (2004), "Implementing matching estimators for average treatment effects in Stata", The Stata Journal, vol. 4, n°3: 290-311.

Abadie A, and Imbens G. W. (2016), "Matching on the estimated propensity score", Econometrica, Vol. 84, n°2: 781-807.

Austin P. C. (2009), "Balance diagnostics for comparing the distribution of baseline covariates between treatment group in propensity-score matched samples", Statistics in Medicine, vol. 28: 3083–3107.

Cattaneo M. D. (2010), "Efficient semiparametric estimation of multi-valued treatment effects under ignorability", Journal of Econometrics, Vol. 155: 138-154.

Stata Treatment-Effects Reference Manual, A Stata Press Publication, Release 14.

Appendix

```
%macro ps_matching(data=,y=,w=,x=,M=,Link=,L=,Lt=);

data table;
set &data(keep=&y &x &w);
_id_+1;
run;

/*** Propensity score estimation ***/

%if %upcase(&Link)=PROBIT %then %let estim_method=PROBIT;%else %let estim_method=LOGIT;

title "Propensity Score Estimation";
proc logistic descending data=table;
model &w = &x/noint Link=&estim_method;
output out=pscore PREDICTED =pscore_pred xbeta=xbeta;
run;
title;

data pscore;
set pscore;
%if %upcase(&Link)=PROBIT %then %do;
BigF=probnorm(xbeta);  /* Probability F(xbeta) */
SmallF=pdf('normal',xbeta,0,1); /* Density function */
%end;
%else %do;
 BigF=exp(xbeta)/(1+exp(xbeta));      /* Probability F(xbeta) */
 Smallf=exp(xbeta)/(1+exp(xbeta))**2; /* Density function */
%end;
run;

/** File with propensity scores for treated **/

data treatment;
set pscore(where=(&w=1) rename=(pscore_pred=pscoreT &y=outcomeT _id_=idT));

/** File with propensity scores for controls **/

data control;
set pscore(where=(&w=0) rename=(pscore_pred=pscoreC  &y=outcomeC _id_=idC));
run;

/* Measuring distance between treated units and control units */

data DistanceT(rename=(idT=_id_ outcomeT=outcome outcomeC=outcomeM idc=idM));
set Treatment(keep=idT pscoreT outcomeT);
do i= 1 to Ncontrol;
 set Control(keep=idC pscoreC outcomeC) point=i nobs=Ncontrol;
 Distance=abs(pscoreT-pscoreC);
 output;
end;
run;
data DistanceC(rename=(idC=_id_ outcomeC=outcome outcomeT=outcomeM idT=idM));
set Control(keep=idC pscoreC outcomeC);
do i= 1 to NTreated;
 set Treatment(keep=idT pscoreT outcomeT) point=i nobs=NTreated;
 Distance=abs(pscoreC-pscoreT);
 output;
end;
run;

/** Closest matches for treated and controls with the Nearest Neighbor Matching Method **/

data distance;
set distanceT distanceC;

proc sort data=distance;
by _id_ distance;

data ClosestJM1 ClosestJM2;
set distance;
by _id_ distance;
retain cardJmi;  /* cardJmi : number of matches for observation i (the number of elements of the set
of matches Jmi) */
if first._id_ then cardJmi=0;
cardJmi=cardJmi+1;
if cardJmi<=&M then output ClosestJM1;else output ClosestJM2;
```

```
/* Dealing with ties */

data tie(drop=cardJmi);
set ClosestJM1(keep=_id_ distance cardJmi rename=(distance=distancetie));
where cardJmi=&M;

data ClosestJM2(drop=distancetie);
merge ClosestJM2 tie;
by _id_;
if distance=distancetie;

data ClosestJM;
set ClosestJM1(drop=cardJmi) ClosestJM2(drop=cardJmi);

proc summary data=ClosestJM nway;
class _id_;
output out=Jm(keep=_id_ _freq_ rename=(_freq_=cardJmi));
run;

proc sort data=ClosestJM;
by _id_ distance;
run;

data ClosestJM;
merge Jm ClosestJM;
by _id_;
run;

/*********************************/
/*** Estimating ATE and ATET ****/
/*********************************/

proc summary data=ClosestJM nway;
class _id_;
var outcomeM;
output out=MeanOutM(keep=_freq_ _id_ Mean_outcomeM rename=(_freq_=cardJmi)) mean=Mean_outcomeM;

data ATE_ATET;
merge MeanOutM table(keep=_id_ &w &y);
by _id_;
N+1;
N1+1*(&w=1);
SumATEi+(2*&w-1)*(&y-Mean_outcomeM);
SumATETi+&w*(&y-Mean_outcomeM);
cst=1;

data ATE_ATET(keep=cst ATE ATET);
set ATE_ATET;
by cst;
if last.cst then do;
 ATE=SumATEi/N;
 ATET=SumATETi/N1;
 output;
end;
run;

/******************************/
/**** Variance estimation ****/
/******************************/

/* 1. Measuring the number of times an observation is used as a match (km_i) */

/* Remark: to account for ties, Km_i is not the number of times that observation i is used as a
match but "the number of times i is used as a match for all observations j of the opposite treatment
group, each time weighted by the total number of matches for observation j" (see Abadie et al.
(2004), "The Stata Journal", vol. 4, n°3, page 293.)  */

proc sort data=ClosestJM;
by idM;

data km(rename=(idM=_id_));
set ClosestJM(keep=idM cardJmi);
by idM;
retain km_i kmprime_i count;
if first.idM then do;
 count=0;
 km_i=0;
 kmprime_i=0;
end;
```

```
count=count+1;  /* count : number of times observation i is used as a match for all observations of
the opposite treatment group */
km_i=km_i+1/cardJmi;
kmprime_i=kmprime_i+1/cardJmi**2;
if last.idM then output;
run;

/** 2. Estimation of the Information matrix , see Abadie and Imbens (2016), page 794 **/

proc iml;
start IMatrix;
use pscore;
read all var {&x} into X;
read all var {BigF} into BigF;
read all var {SmallF} into Smallf;
close pscore;

N=nrow(X);
k=ncol(X);

Ihat=J(k,k,0);
do i=1 to N;
 Ihat=Ihat+(Smallf[i]##2/(BigF[i]#(1-BigF[i])))#t(X[i,])*X[i,];
end;
Ihat_N=Ihat/N;

create Ihat_N from Ihat_N;
append from Ihat_N;

finish IMatrix;
run IMatrix;
quit;

/** 3. Matching treated with treated and controls with controls to detect the sets HLi and HL_i, see
Abadie and Imbens (2016), page 795 **/

/* HLi : "the set of units in the same treatment group as unit i that have the closest L values of
the propensity score to unit i's propensity score, including observation i. HL_i is "similarly
defined but excludes i".*/

data DistanceTT(keep=idT idM distance outcomeM rename=(idT=_id_));
set Treatment(keep=idT pscoreT);
do i= 1 to NTreated;
 set Treatment(keep=idT pscoreT outcomeT rename=(idT=idM pscoreT=pscoreM outcomeT=outcomeM)) point=i
nobs=NTreated;
 Distance=abs(pscoreT-pscoreM);
 output;
end;
run;
data DistanceCC(keep=idC idM distance outcomeM rename=(idC=_id_));
set Control(keep=idC pscoreC);
do i= 1 to NControl;
 set Control(keep=idC pscoreC outcomeC rename=(idC=idM pscoreC=pscoreM outcomeC=outcomeM)) point=i
nobs=NControl;
 Distance=abs(pscoreC-pscoreM);
 output;
end;
run;

data distanceTC;
set distanceTT distanceCC;
run;
proc sort data=distanceTC;
by _id_ distance;

/** Detecting the HLi sets **/

data ClosestHL1 ClosestHL2;
set distanceTC;
by _id_ distance;
retain Li;
if first._id_ then Li=0;
Li=Li+1;
if Li<=&L+1 then output ClosestHL1;else output ClosestHL2;

 /** Dealing with ties **/

data tie(drop=Li);
set ClosestHL1(keep=_id_ distance Li rename=(distance=distancetie));
where Li=&L+1;
```

```
data ClosestHL2(drop=distancetie);
merge ClosestHL2 tie;
by _id_;
if distance=distancetie;

data ClosestHL;
set ClosestHL1(drop=Li) ClosestHL2(drop=Li);
run;

/** Detecting the HL_i sets : HLi without i **/

data distanceTC_;
set distanceTC;
where _id_ ne idM;  /* Excluding unit i from its set */

proc sort data=distanceTC_;
by _id_ distance;

data ClosestHL1_ ClosestHL2_;
set distanceTC_;
by _id_ distance;
retain Li ;
if first._id_ then Li_=0;
Li_=Li_+1;
if Li_<=&L+1 then output ClosestHL1_;else output ClosestHL2_;

 /** Dealing with ties **/

data tie(drop=Li_);
set ClosestHL1_(keep=_id_ distance Li_ rename=(distance=distancetie));
where Li_=&L+1;

data ClosestHL2_(drop=distancetie);
merge ClosestHL2_ tie;
by _id_;
if distance=distancetie;

data ClosestHL_;
set ClosestHL1_(drop=Li_) ClosestHL2_(drop=Li_);

proc sort data=ClosestHL_;
by _id_;

/* 4. Computing the individual variances Sigmabarhat2(Wi,F(Xitetat)), see Abadie and Imbens (2016),
page 796 */

proc summary data=ClosestHL nway vardef=df; /* df : Li-1 at the denominator */
class _id_;
var outcomeM;
output out=sigbarhat(keep=_id_ Sigbarhati) var=Sigbarhati;
run;

/*** 5. Computing sigmahat2 and sigmathat2 ***/

/* Remark : To account for ties, the formulas used below do not correspond exactly to the formulas
of page 795 (Abadie and Imbens, 2016) but to those that appear in Stata Treatment-Effects Reference
manual, release 13, on page 115. */

data sighat;
merge sigbarhat km(keep=_id_ km_i kmprime_i) MeanOutM table(keep=_id_ &w &y);
by _id_;
if km_i=. then km_i=0;
if kmprime_i=. then kmprime_i=0;
cst=1;

data sighat;
merge ATE_ATET sighat;
by cst;
N1+(&w=1);
N+1;
Sumsighat2i+((2*&w-1)*(&y-Mean_outcomeM)-ATE)**2+(km_i**2+2*km_i-kmprime_i)*sigbarhati;
Sumsigthat2i+&w*(&y-Mean_outcomeM-ATET)**2+(1-&w)*(km_i**2-kmprime_i)*sigbarhati;
if last.cst then do;
 sigmahat2=sumsighat2i/N;
 sigmathat2=sumsigthat2i*N/N1**2;
 output;
 end;
run;
```

```
data pscore;
merge km(keep=_id_ count km_i) pscore JM(keep=_id_ cardJmi);
by _id_;
if count=. then count=0;
if km_i=. then km_i=0;
cst=1;
run;

/*** 6. Computing c_hat and ct_hat ***/

/** Detecting JL sets **/

data ClosestJL1 ClosestJL2;
set distance;
by _id_ distance;
retain JLi;  /* Jli : number of elements of the set of matches */
if first._id_ then JLi=0;
JLi=JLi+1;
if JLi<=&L then output ClosestJL1;else output ClosestJL2;

/* Dealing with ties */

data tie(drop=JLi);
set ClosestJL1(keep=_id_ distance JLi rename=(distance=distancetie));
where JLi=&L;

data ClosestJL2(drop=distancetie);
merge ClosestJL2 tie;
by _id_;
if distance=distancetie;

data ClosestJL;
set ClosestJL1(drop=JLi) ClosestJL2(drop=JLi);

proc summary data=ClosestJL nway;
class _id_;
output out=JL(keep=_id_ _freq_ rename=(_freq_=JLi));
run;

proc sort data=ClosestJL;
by _id_ distance;
run;

data ClosestJL;
merge JL ClosestJL;
by _id_;
run;

/** Elements needed for calculating individual covariances between X and y for w=1 and w=0 **/

proc sort data=closestHL;
by idM;

data covarHL;
merge table(keep=_id_ &x rename=(_id_=idM)) closestHL(keep=idM outcomeM _id_ in=a);
by idM;
if a;

proc summary data=covarHL nway ;
class _id_;
var &x outcomeM;
output out=meanXY_HL(drop=_type_ rename=(_freq_=Li)) mean(&x outcomeM)=;

proc sort data=covarHL;
by _id_;
run;

proc sort data=closestJL;
by idM;

data covarJL;
merge table(keep=_id_ &x rename=(_id_=idM)) closestJL(keep=idM outcomeM _id_ in=a);
by idM;
if a;

proc summary data=covarJL nway ;
class _id_;
var &x outcomeM;
output out=meanXY_JL(drop=_type_ rename=(_freq_=Li)) mean(&x outcomeM)=;
```

```
proc sort data=covarJL;
by _id_;
run;
proc sort data=pscore;
by _id_;
run;

proc summary data=closestHL_ nway;
class _id_;
var outcomeM;
output out=meanHL_(keep=_id_ meanoutcomeHL_) mean=meanoutcomeHL_;

proc summary data=closestHL_ nway;
class _id_;
var outcomeM;
output out=meanY_HL_(keep=_id_ meanY_HL_) mean=meanY_HL_;

/** Detecting the matching set on covariates JXL See Abadie and Imbens (2016), page 799 **/

/* Sample covariance matrix of the covariates */

proc corr data=&data cov out=covarianceX(where=(_type_="COV")) vardef=N noprint;
var &x;
run;

/* Measuring distance between treated units and control units */

proc iml;
start distance;
use pscore;
read all var {&x} where(&w=1) into XTreated;
read all var {&y} where(&w=1) into outcomeT;
read all var {_id_} where(&w=1) into idT;
read all var {&x} where(&w=0) into XControl;
read all var {&y} where(&w=0) into outcomeC;
read all var {_id_} where(&w=0) into idC;
close pscore;

Ntreated=nrow(XTreated);
Ncontrol=nrow(XControl);

/** Scaling matrix (Mahalanobis) for measuring the distance between the vector of covariates **/

use covarianceX;
read all var {&x} into S;
close covarianceX;
V=inv(S);

/** For each treated, measuring all distances between its vector of covariates and those of controls
**/

distT=J(Ntreated#Ncontrol,3,0);
do i=1 to Ntreated;
 disti=J(Ncontrol,3,0);
  do l=1 to Ncontrol;
   disti[l,1]=sqrt( (XTreated[i,]-XControl[l,])*V*(XTreated[i,]-XControl[l,])` );
   disti[l,3]=outcomeC[l];
  end;
  disti[,2]=idT[i];
  distT[Ncontrol#(i-1)+1:i#Ncontrol,]=disti;
end;

varnames= {"Distance" "_id_" "OutcomeM"};
create distanceT from distT[colname=varnames ];
append from distT ;

free distT;

/** For each control, measuring all distances between its vector of covariates and those of treated
**/

distC=J(Ncontrol#Ntreated,3,0);
do l=1 to Ncontrol;
 distl=J(Ntreated,3,0);
  do i=1 to Ntreated;
   distl[i,1]=sqrt( (XTreated[i,]-XControl[l,])*V*(XTreated[i,]-XControl[l,])` );
   distl[i,3]=outcomeT[i];
  end;
  distl[,2]=idC[l];
  distC[NTreated#(l-1)+1:l#NTreated,]=distl;
```

```
end;

varnames= {"Distance" "_id_" "OutcomeM"};
create distanceC from distC[colname=varnames];
append from distC;

free distC Xtreated XControl;

finish distance;
run distance;
quit;

/** Closest matches for treated and controls **/

data distance;
set distanceT distanceC;

proc sort data=distance;
by _id_ distance;

data Closest1 Closest2;
set distance;
by _id_ distance;
retain nbelements;
if first._id_ then nbelements=0;
nbelements=nbelements+1;
if nbelements<=&Lt then output Closest1;else output Closest2;

data tie(drop=nbelements);
set Closest1(keep=_id_ distance nbelements rename=(distance=distancetie));
where nbelements=&Lt;

data Closest2(drop=distancetie);
merge Closest2 tie;
by _id_;
if distance=distancetie;

data ClosestJXL;
set Closest1(drop=nbelements) Closest2(drop=nbelements);

proc summary data=ClosestJXL nway;
class _id_;
var outcomeM;
output out=meanY_JXL(keep=_id_ outcomeM) mean=;
run;

/*** Computing adjusted standard errors and editing the results***/

proc iml;
start cov_editing;
use covarHL;
read all var {&x} into X_HL;
read all var {outcomeM} into y_HL;
close covarHL;
use meanXY_HL;
read all var {Li} into L_HL;
read all var {&x} into meanX_HL;
read all var {outcomeM} into meanY_HL;
close meanXY_HL;
use covarJL;
read all var {&x} into x_JL;
read all var {outcomeM} into y_JL;
close covarJL;
use meanXY_JL;
read all var {Li} into L_JL;
read all var {&x} into meanX_JL;
read all var {outcomeM} into meanY_JL;
close meanXY_JL;
use meanY_HL_;
read all var {meanY_HL_} into meanY_HL_;
close meanY_HL_;

covHL=J(nrow(L_HL),ncol(X_HL),0);
Li_1=0;
do i=1 to nrow(L_HL);
 Li=L_HL[i];
 meanXi=meanX_HL[i,]@J(Li,1,1);
 meanYi=meanY_HL[i,]@J(Li,1,1);
 covi=(x_HL[Li_1+1:Li_1+Li,]-meanXi)#(y_HL[Li_1+1:Li_1+Li,]-meanYi);  /** See Abadie and Imbens
(2016), page 797 **/
```

```
 covHL[i,]=covi[+,]/(Li-1);
 Li_1=Li_1+Li;
end;

covJL=J(nrow(L_JL),ncol(X_JL),0);
Li_1=0;
do i=1 to nrow(L_JL);
 Li=L_JL[i];
 meanXi=meanX_JL[i,]@J(Li,1,1);
 meanYi=meanY_JL[i,]@J(Li,1,1);
 covi=(x_JL[Li_1+1:Li_1+Li,]-meanXi)#(y_JL[Li_1+1:Li_1+Li,]-meanYi);  /** See Abadie and Imbens
(2016), page 797 **/
 covJL[i,]=covi[+,]/(Li-1);
 Li_1=Li_1+Li;
end;

free meanXi meanYi covi Li Li_1 x_HL y_HL x_JL y_JL L_HL L_JL meanX_HL meanX_JL;

use pscore;
read all var {Smallf} into Smallf;
read all var {BigF} into Bigf;
read all var {&w} into w;
read all var {&x} into x;
read all var {&y} into Y;
read all var {count} into countT where(&w=1 & count>0);
read all var {count} into countC where(&w=0 & count>0);
close pscore;

N=nrow(w);
N1=w[+,];
N0=N-N1;

use ATE_ATET;
read all var {ATET} into ATET;
read all var {ATE} into ATE;
close ATE_ATET;

use meanY_JXL;
read all var {outcomeM} into meanY_JXL;
close meanY_JXL;

ci_hat=( (w#covHL+(1-w)#covJL)/BigF + (w#covJL+(1-w)#covHL)/(1-BigF) )#Smallf;
c_hat=ci_hat[+,]/N; /** See Abadie and Imbens (2016), page 797 **/

mu0=w#meanY_JL+(1-w)#meanY_HL_; /** See Abadie and Imbens (2016), page 798 **/
mu1=w#meanY_HL_+(1-w)#meanY_JL;

ct1i_hat=x#Smallf#(mu1-mu0-ATET); /** See Abadie and Imbens (2016), page 798 **/
ct1_hat=ct1i_hat[+,]/N1;

ct2i_hat=( w#covHL+(1-w)#covJL + (w#covJL+(1-w)#covHL)#BigF/(1-BigF) )#Smallf;
ct2_hat=ct2i_hat[+,]/N1; /** See Abadie and Imbens (2016), page 798 **/

ct_hat=ct1_hat+ct2_hat;

free mu0 mu1 ct1i_hat ct2i_hat;

firstderivi=x#Smallf#((2#w-J(nrow(w),1,1))#(Y-meanY_JXL)-ATET); /* See Abadie and Imbens (2016),
page 799 */
firstderiv=firstderivi[+,]/N1;

use sighat;
read all var {sigmahat2} into sigmahat2;
read all var {sigmathat2} into sigmathat2;
close sighat;
use Ihat_N;
read all var _all_ into Ihat_N;
close Ihat_N;

invI=inv(Ihat_N); free Ihat_N;

var_adj=(sigmahat2-c_hat*invI*c_hat`)/N ; /* See Abadie and Imbens (2016), page 799 */
stderr_adj=sqrt(var_adj);

vart_adj=(sigmathat2-ct_hat*invI*ct_hat`+firstderiv*invI*firstderiv`)/N;
stderrt_adj=sqrt(vart_adj);


coef=ATE//ATET;
StdErr=stderr_adj//stderrt_adj;
```

```
z=coef/stderr;

pvalue=round(2*(1-probnorm(abs(z))),.001);

Lbound=round(z-quantile('normal',.975)#StdErr,.001);
Ubound=round(z+quantile('normal',.975)#StdErr,.001);

coef_=round(coef,.001);
z_=round(z,.001);
StdErr_=round(StdErr,.001);

nbmatchesT=nrow(countT);
nbmatchesC=nrow(countC);

maxT=countT[<>];
minT=countT[><];
maxC=countC[<>];
minC=countC[><];

ModelSpecification={"&y","&w","&x"};
Results=coef_||StdErr_||z_||pvalue||LBound||UBound;
SummaryStatistics=N//N0//N1//nbmatchesT//nbmatchesC//minT//maxT//minC//maxC;

TitleResult={"Estimate","Std.Error","z","P-value","L. bound 95% CI","U. bound 95% CI"};
TitleModel={"Outcome variable:","Binary treatment:","Matching variables:"};
TitleOptions={"Number of matches requested (M):","Number of matches requested in the same treatment
group (L and Lt):"};
TitleModel={"Outcome variable:","Binary treatment:","Matching variables:"};

TitleSummary={"Number of observations:","Number of control units:","Number of treated units:",
              "Number of treated units matched to controls:",
                         "Number of control units matched to treated:",
                  "Number of times a treated unit is used as a match (MIN):",
                  "Number of times a treated unit is used as a match (MAX):",
                  "Number of times a control unit is used as a match (MIN):",
                  "Number of times a control unit is used as a match (MAX):"};

EstimationOptions={"&M"}//{"&L and &Lt"};
Effect={"Average Treatment Effect (ATE)","Average Treatment Effect for the Treated (ATET)"};

print "ESTIMATING AVERAGE TREATEMENT EFFECTS with PROPENSITY SCORE MATCHING",,
      ModelSpecification [rowname=TitleModel],,
      EstimationOptions [rowname=TitleOptions],,
      SummaryStatistics [rowname=TitleSummary],,
         "Estimation results with Abadie and Imbens (2016) robust standard errors",
      Results [colname=TitleResult rowname=Effect];
finish cov_editing;
run cov_editing;
quit;

/**** Checking common support, visual diagnostic ****/

proc sort data=pscore;
by &w;

ODS GRAPHICS OFF;
TITLE;
proc format;
value treat
      0="Control"
      1="Treated";
run;
proc boxplot data=pscore;
plot (pscore_pred)*&w/ nohlabel;
title "Common support: visual diagnostic";
format &w treat.;
run;

/*** Checking balance : standardized differences for comparing means between controls and treated
before and after matching. Formulas for "continuous" and binary variables are different (see Austin,
P. C. (2009)). ***/

/* Identifying binary and continuous covariates */

ods select nlevels;
ods table nlevels=n_levels;
proc freq data=table nlevels;
table &x /noprint;
title "Identifying continuous covariates";
```

```
run;

title;

data n_levels;
set n_levels end=end;
contvar+(nlevels>2); /* if more than 2 categories then "continuous", else binary */
binaryvar+(nlevels=2);
if end then call symputx('nbcont',contvar);
if end then call symputx('nbbinary',binaryvar);
run;

proc iml;
start balance;

use pscore;
read all var {&x} where(&w=0) into XC;
read all var {&x} where(&w=1) into XT;
read all var {&x} where(&w=0 & km_i>0) into XCmatched; /* covariates for controls used as matches
for treated */
read all var {&x} where(&w=1 & km_i>0) into XTmatched; /* covariates for treated used as matches for
controls */
close pscore;

use n_levels;
read all var {Nlevels} into Nlevels;
close n_levels;

binary=(Nlevels=2);
k=ncol(xc);

MeanXC=XC[:,];
MeanXT=XT[:,];

meanXCmatched=XCmatched[:,];
meanXTmatched=XTmatched[:,];

VarXC=Var(XC);
VarXT=Var(XT);

VarXCmatched=Var(XCmatched);
VarXTmatched=Var(XTmatched);

 /** Different formulas for continuous and binary variables **/

Norm_Diff_Before=J(k,1,0);
Norm_Diff_TCmatches=J(k,1,0);
Norm_Diff_CTmatches=J(k,1,0);

do i=1 to k;
 if binary[i]=0 then do;
  Norm_Diff_Before[i]=round( t((meanXT[,i]-meanXC[,i])/sqrt((varXT[,i]+varXC[,i])/2)) ,.001);
  Norm_Diff_TCmatches[i]=round( t((meanXT[,i]-
meanXCmatched[,i])/sqrt((varXT[,i]+varXCmatched[,i])/2)) ,.001);
  Norm_Diff_CTmatches[i]=round( t((meanXTmatched[,i]-
meanXC[,i])/sqrt((varXTmatched[,i]+varXC[,i])/2)) ,.001);
 end;
 else do;
  Norm_Diff_Before[i]=round( t( (meanXT[,i]-meanXC[,i])/sqrt( (meanXT[,i]#(1-
meanXT[,i])+meanXC[,i]#(1-meanXC[,i]))/2 ) ) ,.001);
  Norm_Diff_TCmatches[i]=round( t( (meanXT[,i]-meanXCmatched[,i])/sqrt( (meanXT[,i]#(1-
meanXT[,i])+meanXCmatched[,i]#(1-meanXCmatched[,i]))/2 ) ) ,.001);
  Norm_Diff_CTmatches[i]=round( t( (meanXTmatched[,i]-meanXC[,i])/sqrt( (meanXTmatched[,i]#(1-
meanXTmatched[,i])+meanXC[,i]#(1-meanXC[,i]))/2 ) ) ,.001);
 end;
end;

Normalized_Differences=Norm_Diff_Before||Norm_Diff_TCmatches||Norm_Diff_CTmatches;

Variables={&x}`;
Title={"Unmatched Sample" "Matched sample (T)" "Matched sample (C)"};
print Normalized_Differences [colname=Title rowname=Variables label="Normalized covariate mean
differences between treated and controls"],,
      "Note: 'Matched sample (T)' is for normalized mean differences between all sample treated and
their matches, 'Matched sample (C)' for
       normalized mean differences between all sample controls and their matches.";
finish balance;
run balance;

/** Checking balance : contingency tables and visual diagnostics **/
```

```
 proc means data=pscore noprint;
 var pscore_pred;
 output out=statscore(drop=_type_ _freq_) P20= P40= P60= P80= /autoname;
 data statscore;
 set statscore;
 cst=1;
 data pscore;
 merge statscore pscore;
 by cst;
 interval=1*(pscore_pred<pscore_pred_P20)+
          2*(pscore_pred_P20<=pscore_pred<pscore_pred_P40)+
                 3*(pscore_pred_P40<=pscore_pred<pscore_pred_P60)+
                 4*(pscore_pred_P60<=pscore_pred<pscore_pred_P80)+
          5*(pscore_pred>=pscore_pred_P80);
 run;

 data sampleT0;
 set pscore(keep=&w &x interval where=(&w=1));
 SampleType=1;
 data sampleC0;
 set pscore(keep=&w &x interval where=(&w=0));
 SampleType=2;
 data sampleTmatched;
 set pscore(keep=&w &x km_i interval where=(&w=1 and km_i>0));
 SampleType=3;
 data sampleCmatched;
 set pscore(keep=&w &x km_i interval where=(&w=0 and km_i>0));
 SampleType=4;

 data sample;
 set sampleT0 sampleC0 sampleTmatched sampleCmatched;
 proc sort data=sample;
 by SampleType;
 run;

 proc format;
 value sample
      1="All treated"
          2="All controls"
          3="Matched controls"
          4="Matched treated";
 run;
/* Identifying the continuous variables and picking up their names */
/* See http://support.sas.com/kb/45/626.html "Using Macro to create new variable names from variable
values" */

%if &nbbinary>0 %then %do;
 data binaryvar;
 set n_levels(where=(nlevels=2));
 run;
 data _null_;
 set binaryvar end=end;
 count+1;
 call symputx('varb'||left(count),TableVar);
 if end then call symputx('maxb',count);
 run;
 %macro binaryvars;
 data binaryvar;
 set binaryvar end=end;
 %do i = 1 %to &maxb;
  if _n_=&i then do;
    &&varb&i=0;
    retain &&varb&i;
    keep &&varb&i;
  end;
 %end;
 if end then output;
 %mend;

 %binaryvars;

 proc contents data=binaryvar out=binaryvar(keep =varnum name) noprint;
 run;
 proc sql noprint;
 select name
 into :xbinary separated by " " from binaryvar order by varnum;
 quit;
 %do i=1 %to 5;
```

```
  title "Comparing binary covariate distributions between treatment groups before and after
matching";
  title2 "Interval &i of the estimated propensity score distribution";
  proc tabulate data=sample(where=(interval=&i));
  class sampletype &xbinary;
  table (&xbinary)*COLPCTN,sampletype;
  keylabel COLPCTN='Distribution in %';
  format sampletype sample.;
  run;
 %end;
%end;
%if &nbcont>0 %then %do;
 data contvar;
 set n_levels(where=(nlevels>2));
 run;
 data _null_;
 set contvar end=end;
 count+1;
 call symputx('var'||left(count),TableVar);
 if end then call symputx('max',count);
 run;
 %macro contvars;
 data contvar;
 set contvar end=end;
 %do i = 1 %to &max;
  if _n_=&i then do;
    &&var&i=0;
    retain &&var&i;
    keep &&var&i;
  end;
 %end;
 if end then output;
 %mend;

 %contvars;

 proc contents data=contvar out=contvar(keep =varnum name) noprint;
 run;
 proc sql noprint;
 select name
 into :xcont separated by " " from contvar order by varnum;
 quit;
 %do i=1 %to 5;
  ODS GRAPHICS OFF;
  TITLE;
  proc boxplot data=sample(where=(interval=&i));
  plot (&xcont)*sampleType/ nohlabel;
  title "Checking balance through box-plots, interval &i of the estimated propensity score
distribution";
  format sampleType sample.;
  run;
  ODS GRAPHICS ON;
  proc npar1way edf plots(nostats)=edfplot data=sample(where=(interval=&i and sampleType in (1,2)));
  class sampleType ;
  var &xcont;
  title "Checking balance through empirical distribution functions, interval &i of the estimated
propensity score distribution";
  title2 "Original sample of treated vs original sample of controls";
  format sampleType sample.;
  run;
  proc npar1way edf plots(nostats)=edfplot data=sample(where=(interval=&i and sampleType in (1,3)));
  class sampleType ;
  var &xcont;
  title "Checking balance through empirical distribution functions, interval &i of the estimated
propensity score distribution";
  title2 "Original sample of treated vs final sample of matched controls";
  format sampleType sample.;
  run;
  proc npar1way edf plots(nostats)=edfplot data=sample(where=(interval=&i and sampleType in (2,4)));
  class sampleType ;
  var &xcont ;
  title "Checking balance through empirical distribution functions, interval &i of the estimated
propensity score distribution";
  title2 "Original sample of controls vs final sample of matched treated";
  format sampleType sample.;
  run;
  quit;
  title;
 %end;
%end;
```

```
/** Output data files **/

data outdata1;
set pscore(keep=_id_ &y &w &x km_i count CardJMi pscore_pred);

data outdata2;
set closestJM(keep=_id_ idM outcomeM distance);
run;
%mend ps_matching;
```