The Stata Journal (2002)
2, Number 4, pp. 358–377

Estimation of average treatment effects based on propensity scores

Sascha O. Becker
University of Munich

Andrea Ichino
EUI

Abstract. In this paper, we give a short overview of some propensity score matching estimators suggested in the evaluation literature, and we provide a set of Stata programs, which we illustrate using the National Supported Work (NSW) demonstration widely known in labor economics.

Keywords: st0026, propensity score, matching, average treatment effect, evaluation

1 Introduction

In the evaluation literature, data often do not come from randomized trials but from (nonrandomized) observational studies. In seminal work, Rosenbaum and Rubin (1983) proposed propensity score matching as a method to reduce the bias in the estimation of treatment effects with observational datasets. These methods have become increasingly popular in medical trials and in the evaluation of economic policy interventions.

Since in observational studies assignment of subjects to the treatment and control groups is not random, the estimation of the effect of treatment may be biased by the existence of confounding factors. Propensity score matching is a way to "correct" the estimation of treatment effects controlling for the existence of these confounding factors based on the idea that the bias is reduced when the comparison of outcomes is performed using treated and control subjects who are as similar as possible. Since matching subjects on an n-dimensional vector of characteristics is typically unfeasible for large n, this method proposes to summarize pretreatment characteristics of each subject into a single-index variable (the propensity score) that makes the matching feasible.

In this paper, we give a short overview of some propensity score matching estimators suggested in the evaluation literature, and we provide a set of Stata programs, which we illustrate using the National Supported Work (NSW) demonstration widely known in labor economics. In using these programs, it should be kept in mind that they only allow to reduce, and not to eliminate, the bias generated by unobservable confounding factors. The extent to which this bias is reduced depends crucially on the richness and quality of the control variables on which the propensity score is computed and the matching performed. To be more precise, the bias is eliminated only if the exposure to treatment can be considered to be purely random among individuals who have the same value of the propensity score.

2 The propensity score

The propensity score is defined by Rosenbaum and Rubin (1983) as the conditional probability of receiving a treatment given pretreatment characteristics:

$$p(X) \equiv \Pr(D = 1|X) = E(D|X) \tag{1}$$

where $D = \{0, 1\}$ is the indicator of exposure to treatment and X is the multidimensional vector of pretreatment characteristics. Rosenbaum and Rubin (1983) show that if the exposure to treatment is random within cells defined by X, it is also random within cells defined by the values of the one-dimensional variable $p(X)$. As a result, given a population of units denoted by i, if the propensity score $p(X_i)$ is known, then the Average effect of Treatment on the Treated (ATT) can be estimated as follows:

$$\begin{aligned} \tau &\equiv E\{Y_{1i} - Y_{0i}|D_i = 1\} \\ &= E\big[E\{Y_{1i} - Y_{0i}|D_i = 1, p(X_i)\}\big] \\ &= E\big[E\{Y_{1i}|D_i = 1, p(X_i)\} - E\{Y_{0i}|D_i = 0, p(X_i)\}|D_i = 1\big] \end{aligned} \tag{2}$$

where the outer expectation is over the distribution of $(p(X_i)|D_i = 1)$ and Y_{1i} and Y_{0i} are the potential outcomes in the two counterfactual situations of (respectively) treatment and no treatment.

Formally, the following two hypotheses are needed to derive (2) given (1).[1]

Lemma 1 Balancing of pretreatment variables given the propensity score.
If $p(X)$ is the propensity score, then

$$D \quad \perp \quad X \quad | \quad p(X)$$

Lemma 2 Unconfoundedness given the propensity score.
Suppose that assignment to treatment is unconfounded; i.e.,

$$Y_1, Y_0 \quad \perp \quad D \quad | \quad X$$

Then assignment to treatment is unconfounded given the propensity score, i.e.,

$$Y_1, Y_0 \quad \perp \quad D \quad | \quad p(X)$$

If the Balancing Hypothesis of Lemma 1 is satisfied, observations with the same propensity score must have the same distribution of observable (and unobservable) characteristics independently of treatment status. In other words, for a given propensity score, exposure to treatment is random and therefore treated and control units should be on average observationally identical. Any standard probability model can be used to estimate the propensity score. For example, $\Pr(D_i = 1|X_i) = F\{h(X_i)\}$, where $F(.)$ is the normal or the logistic cumulative distribution and $h(X_i)$ is a function of covariates with linear and higher order terms. The choice of which higher order terms to include

[1] See Rosenbaum and Rubin (1983) or Imbens (2000) for a proof.

is determined solely by the need to obtain an estimate of the propensity score that satisfies the Balancing Hypothesis. Inasmuch as the specification of $h(X_i)$ that satisfies the Balancing Hypothesis is more parsimonious than the full set of interactions needed to match cases and controls on the basis of observables, the propensity score reduces the dimensionality problem of matching treated and control units on the basis of the multidimensional vector X.[2]

The program `pscore.ado` estimates the propensity score and tests the Balancing Hypothesis (Lemma 1) according to the following algorithm:[3]

1. Fit the probit (or logit) model:

   $$\Pr(D_i = 1|X_i) = \Phi\{h(X_i)\}$$

   where Φ denotes the normal (logistic) c.d.f. and $h(X_i)$ is a starting specification that includes all the covariates as linear terms without interactions or higher order terms.

2. Split the sample into k equally spaced intervals of the propensity score, where k is determined by the user and the default is 5.

3. Within each interval, test that the average propensity score of treated and control units does not differ.

4. If the test fails in one interval, split the interval in half and test again.

5. Continue until, in all intervals, the average propensity score of treated and control units does not differ.

6. Within each interval, test that the means of each characteristic do not differ between treated and control units. This is a necessary condition for the Balancing Hypothesis.[4]

7. If the means of one or more characteristics differ, inform the user that the balancing property is not satisfied and that a less parsimonious specification of $h(X_i)$ is needed.

Steps 2–7 of the algorithm can be restricted to the common support. This restriction implies that the test of the balancing property is performed only on the observations whose propensity score belongs to the intersection of the supports of the propensity score of treated and controls. Imposing the common support condition in the estimation of the propensity score may improve the quality of the matches used to estimate the ATT.[5]

[2]It is important to note that the outcome plays no role in the algorithm for the estimation of the propensity score. This is equivalent, in this context, to what happens in controlled experiments in which the design of the experiment has to be specified independently of the outcome.

[3]Note that the Unconfoundedness Hypothesis of Lemma 2 cannot be tested.

[4]Note that it is not sufficient in the sense that the balancing may not hold for higher order moments of the distribution of characteristics. So, to be precise, the program does not test the Balancing Hypothesis, but only one of its implications. In future versions of the program we plan to add tests for higher moments of the distribution of characteristics.

[5]See the next section for further discussion of the common support condition.

3 Matching estimators of the ATT based on the propensity score

An estimate of the propensity score is not enough to estimate the ATT of interest using (2). The reason is that the probability of observing two units with exactly the same value of the propensity score is in principle zero since $p(X)$ is a continuous variable. Various methods have been proposed in the literature to overcome this problem, and four of the most widely used are *Nearest-Neighbor Matching*, *Radius Matching*, *Kernel Matching*, and *Stratification Matching*.

Beginning with the latter, the *Stratification* method consists of dividing the range of variation of the propensity score in intervals such that within each interval, treated and control units have on average the same propensity score. For practical purposes the same blocks identified by the algorithm that estimates the propensity score can be used. Then, within each interval in which both treated and control units are present, the difference between the average outcomes of the treated and the controls is computed. The ATT of interest is finally obtained as an average of the ATT of each block with weights given by the distribution of treated units across blocks.

One of the pitfalls of the Stratification method is that it discards observations in blocks where either treated or control units are absent. This observation suggests an alternative way to match treated and control units, which consists of taking each treated unit and searching for the control unit with the closest propensity score; i.e., the *Nearest Neighbor*. Although it is not necessary, the method is usually applied *with replacement*, in the sense that a control unit can be a best match for more than one treated unit. Once each treated unit is matched with a control unit, the difference between the outcome of the treated units and the outcome of the matched control units is computed. The ATT of interest is then obtained by averaging these differences.

While in the Stratification method, there may be treated units that are discarded because no control is available in their block, in the Nearest-Neighbor method, all treated units find a match. However, it is obvious that some of these matches are fairly poor because for some treated units the nearest neighbor may have a very different propensity score, and, nevertheless, he would contribute to the estimation of the treatment effect independently of this difference. The *Radius Matching* and *Kernel Matching* methods offer a solution to this problem. With *Radius Matching*, each treated unit is matched only with the control units whose propensity score falls into a predefined neighborhood of the propensity score of the treated unit. If the dimension of the neighborhood (i.e., the radius) is set to be very small, it is possible that some treated units are not matched because the neighborhood does not contain control units. On the other hand, the smaller the size of the neighborhood, the better the quality of the matches. With *Kernel Matching*, all treated are matched with a weighted average of *all* controls with weights that are inversely proportional to the distance between the propensity scores of treated and controls.

It is clear from the above considerations that these four methods reach different points on the frontier of the trade-off between quality and quantity of the matches, and

none of them is *a priori* superior to the others. Their joint consideration, however, offers a way to assess the robustness of the estimates.

It should also be noted that with all these methods, the quality of the matches may be improved by imposing the common support restriction. Note, however, that in this way high quality matches may be lost at the boundaries of the common support and the sample may be considerably reduced, so imposing the common support restriction is not necessarily better (see Lechner 2001). All of our programs allow for the common support option as discussed below.

We now proceed to a more detailed and formal description of these estimators. We start with the joint analysis of *Nearest-Neighbor Matching* and *Radius Matching* that can be described in a common framework, moving next to *Kernel Matching* and *Stratification Matching*.

Nearest-Neighbor Matching (attnd.ado and attnw.ado) and Radius Matching (attr.ado)

Let T be the set of treated units and C the set of control units, and let Y_i^T and Y_j^C be the observed outcomes of the treated and control units, respectively. Denote by $C(i)$ the set of control units matched to the treated unit i with an estimated value of the propensity score of p_i. Nearest-neighbor matching sets

$$C(i) = \min_j \parallel p_i - p_j \parallel$$

that is a singleton set unless there are multiple nearest neighbors. In practice, the case of multiple nearest neighbors should be very rare, in particular if the set of characteristics X contains continuous variables. The likelihood of multiple nearest neighbors is further reduced if the propensity score is estimated and saved in double precision.

In radius matching,

$$C(i) = \{p_j \mid \quad \parallel p_i - p_j \parallel < r\}$$

i.e., all the control units with estimated propensity scores falling within a radius r from p_i are matched to the treated unit i.

Both nearest neighbor and radius matching denote the number of controls matched with observation $i \in T$ by N_i^C and define the weights $w_{ij} = \frac{1}{N_i^C}$ if $j \in C(i)$ and $w_{ij} = 0$ otherwise. Then, the formula for both types of matching estimators can be written as follows:

$$\begin{aligned}
\tau^M &= \frac{1}{N^T}\sum_{i\in T}\left(Y_i^T - \sum_{j\in C(i)} w_{ij}Y_j^C\right) \\
&= \frac{1}{N^T}\left(\sum_{i\in T} Y_i^T - \sum_{i\in T}\sum_{j\in C(i)} w_{ij}Y_j^C\right) \\
&= \frac{1}{N^T}\sum_{i\in T} Y_i^T - \frac{1}{N^T}\sum_{j\in C} w_j Y_j^C
\end{aligned}$$

(where M stands for either nearest-neighbor matching or radius matching, and the number of units in the treated group is denoted by N^T): where the weights w_j are defined by $w_j = \Sigma_i w_{ij}$.

To derive the variances of these estimators, the weights are assumed to be fixed and the outcomes are assumed to be independent across units.

$$\begin{aligned}
\mathrm{Var}(\tau^M) &= \frac{1}{(N^T)^2}\left\{\sum_{i\in T}\mathrm{Var}(Y_i^T) + \sum_{j\in C}(w_j)^2\mathrm{Var}(Y_j^C)\right\} \\
&= \frac{1}{(N^T)^2}\left\{N^T\mathrm{Var}(Y_i^T) + \sum_{j\in C}(w_j)^2\mathrm{Var}(Y_j^C)\right\} \\
&= \frac{1}{N^T}\mathrm{Var}(Y_i^T) + \frac{1}{(N^T)^2}\sum_{j\in C}(w_j)^2\mathrm{Var}(Y_j^C)
\end{aligned}$$

In the programs `attnd.ado`, `attnw.ado`, and `attr.ado`, standard errors are obtained analytically using the above formula or by bootstrapping using the `bootstrap` option.

The difference between `attnd.ado` and `attnw.ado` is most easily understood by briefly describing the way nearest neighbors are computationally determined in the two programs. To save on computing time, nearest neighbors are not determined by comparing treated observations to every single control, but rather by first sorting all records by the estimated propensity score, and then searching forward and backward for the closest control unit(s). If, for a treated unit, forward and backward matches happen to be equally good, there are two computationally feasible options to obtain analytical standard errors while at the same time exploiting the very fast forward and backward search strategy: `attnw.ado` gives equal weight (hence, the letters "nw" for *n*earest neighbor and equal *w*eight) to the groups of forward and backward matches; `attnd.ado` randomly draws either the forward or backward matches (hence, the letters "nd" for *n*earest neighbor and random *d*raw). In practice, the case of multiple nearest neighbors should be very rare, especially if the set of Xs contains continuous variables, in which case both `attnw.ado` and `attnd.ado` should give equal results. The likelihood of multiple nearest neighbors is further reduced if the propensity score is estimated and saved in double precision, which is what `pscore.ado` does by default.

Kernel matching method (attk.ado)

The kernel matching estimator is given by

$$\tau^K = \frac{1}{N^T} \sum_{i \in T} \left\{ Y_i^T - \frac{\sum_{j \in C} Y_j^C G(\frac{p_j - p_i}{h_n})}{\sum_{k \in C} G(\frac{p_k - p_i}{h_n})} \right\}$$

where $G(\cdot)$ is a kernel function and h_n is a bandwidth parameter. Under standard conditions on the bandwidth and kernel,

$$\frac{\sum_{j \in C} Y_j^C G(\frac{p_j - p_i}{h_n})}{\sum_{k \in C} G(\frac{p_k - p_i}{h_n})}$$

is a consistent estimator of the counterfactual outcome Y_{0i}. In the program `attk.ado`, standard errors are obtained by bootstrapping using the `bootstrap` option. Users can choose the default Gaussian kernel or the Epanechnikov kernel.

Stratification method (atts.ado)

This method is based on the same stratification procedure used for estimating the propensity score. Note that by construction in each block defined by this procedure, the covariates are balanced and the assignment to treatment can be considered random. Hence, letting q index the blocks defined over intervals of the propensity score, within each block the program computes

$$\tau_q^S = \frac{\sum_{i \in I(q)} Y_i^T}{N_q^T} - \frac{\sum_{j \in I(q)} Y_j^C}{N_q^C}$$

where $I(q)$ is the set of units in block q and N_q^T and N_q^C are the numbers of treated and control units in block q.

The estimator of the ATT in (2) based on the Stratification method is then computed using the formula

$$\tau^S = \sum_{q=1}^{Q} \tau_q^S \frac{\sum_{i \in I(q)} D_i}{\sum_{\forall i} D_i}$$

where the weight for each block is given by the corresponding fraction of treated units and Q is the number of blocks.

Assuming independence of outcomes across units, the variance of τ^S is

$$\mathrm{Var}(\tau^S) = \frac{1}{N^T} \left\{ \mathrm{Var}(Y_i^T) + \sum_{q=1}^{Q} \frac{N_q^T}{N^T} \frac{N_q^T}{N_q^C} \mathrm{Var}(Y_j^C) \right\}$$

In the program `atts.ado`, standard errors are obtained analytically using the above formula, or by bootstrapping using the `bootstrap` option.

4 Syntax

pscore *and* att* *are regression-like commands.*

pscore *treatment* [*varlist*] [*weight*] [if *exp*] [in *range*] , pscore(*newvar*) [blockid(*newvar*) detail logit comsup level(#) numblo(#)]

attnd *outcome treatment* [*varlist*] [*weight*] [if *exp*] [in *range*] [, pscore(*scorevar*) logit index comsup detail bootstrap reps(#) noisily dots]

attnw *outcome treatment* [*varlist*] [*weight*] [if *exp*] [in *range*] [, pscore(*scorevar*) logit index comsup detail bootstrap reps(#) noisily dots]

attr *outcome treatment* [*varlist*] [*weight*] [if *exp*] [in *range*] [, pscore(*scorevar*) logit index radius(#) comsup detail bootstrap reps(#) noisily dots]

attk *outcome treatment* [*varlist*] [*weight*] [if *exp*] [in *range*] [, pscore(*scorevar*) logit index epan bwidth(#) comsup detail bootstrap reps(#) noisily dots]

atts *outcome treatment* [*varlist*] [if *exp*] [in *range*] , pscore(*scorevar*) blockid(*blockvar*) [comsup detail bootstrap reps(#) noisily dots]

fweights, iweights, and pweights are allowed with all commands except atts; see [U] **14.1.6 weight**. No weights are allowed with atts.

5 Options

Note the following points:

- It is important to clean up your dataset before running this suite of Stata programs, in particular to delete observations with missing values.
- In pscore, the option pscore(*newvar*) is compulsory.
- pscore and the att* programs are closely related in that users will typically first run pscore to estimate the propensity score and test whether the balancing property holds, and then proceed to estimate the ATT with one or more of the att* programs.

- Note, however, that all `att*` programs are stand-alone programs; i.e., if users prefer to estimate the propensity score with their own procedure, they can do so, specifying the name of the estimated propensity score as an input variable in the `att*` programs.

- If users are confident about the "correct" specification for the propensity score, then that specification can be used directly in the `att*` programs to estimate the propensity score without first running `pscore`. This is actually advisable if bootstrapped standard errors are requested because, when users do not specify their own previously estimated propensity score, the bootstrap encompasses the estimation of the propensity score based on the specification given by *varlist*. Re-estimating the propensity score at each replication of the bootstrap procedure is recommended to account for the uncertainty associated with the estimation of the propensity score. This is especially true when the `comsup` option is specified, because in this case the region of common support changes with every bootstrap sample, and bootstrapped standard errors pick up this uncertainty as well. So, typically, users would first identify a specification satisfying the balancing property—using, for example, `pscore`—and then provide exactly this specification in *varlist* and use bootstrapped standard errors.

- For `atts`, in addition to the estimated propensity score, users must provide a variable containing the block identifier for the estimated propensity score. Therefore, in the case of `atts`, it is most convenient to rely on `pscore`, because it nicely generates both the estimated propensity score variable and the block identifier.

5.1 Options for pscore

`pscore(`*newvar*`)` is a compulsory option and specifies the variable name for the estimated propensity score.

`blockid(`*newvar*`)` specifies the variable name for the block number of the estimated propensity score.

`detail` requests that more detailed output documenting the steps performed to obtain the final results be displayed.

`logit` specifies that a logit model to estimate the propensity score be used instead of the default probit model.

`comsup` restricts the analysis of the balancing property to all treated plus those controls in the region of common support. A dummy variable named `comsup` is added to the dataset to identify the observations in the common support.

`level(#)` specifies the significance level of the tests of the balancing property. The default is 0.01. Note that this significance level applies to the test of each single variable of the vector X of pretreatment characteristics; i.e., the balancing property is not rejected only in the case that it holds for every single X. This is a relatively conservative approach because of the following argument. Assume that the significance level is set to 0.05, that X consists of 20 variables, and that the

tests of the balancing property are mutually independent. Then, with probability $\binom{20}{1}(0.05)^1(0.95)^{19} = 0.37$, one of the tests rejects the balancing property although it actually holds true.

`numblo(#)` specifies the number of blocks of equal score range to be used at the beginning of the test of the balancing hypothesis. The default is 5 blocks.

5.2 Options common to all att* commands

`comsup` restricts the computation of the ATT to the region of common support.

`detail` requests that more detailed output documenting the steps performed to obtain the final results be displayed.

`bootstrap` bootstraps the standard error of the treatment effect.

`reps(#)` specifies the number of bootstrap replications to be performed. The default is 50. This option produces an effect only if the `bootstrap` option is specified.

`noisily` requests that any output from the bootstrap replications be displayed. This option produces an effect only if the `bootstrap` option is specified.

`dots` requests that a dot be placed on the screen at the beginning of each bootstrap replication. This option produces an effect only if the `bootstrap` option is specified.

5.3 Options for attnd and attnw

`pscore(`*scorevar*`)` specifies the name of the user-provided variable containing the estimated propensity score. If this option is not specified, `attnd` and `attnw` will estimate the propensity score with the specification provided in *varlist* using a probit.

`logit` requests that a logit estimation of the propensity score be used instead of the default probit model when the option `pscore(`*scorevar*`)` is not specified by the user.

`index` requires the use of the linear index as the propensity score when the option `pscore(`*scorevar*`)` is not specified by the user.

5.4 Options for attr

`pscore(`*scorevar*`)` specifies the name of the user-provided variable containing the estimated propensity score. If this option is not specified, `attr` will estimate the propensity score with the specification provided in *varlist* using a probit.

`logit` requests that a logit estimation of the propensity score be used instead of the default probit model when the option `pscore(`*scorevar*`)` is not specified by the user.

`index` requires the use of the linear index as the propensity score when the option `pscore(`*scorevar*`)` is not specified by the user.

`radius(#)` specifies the size of the radius. The default is 0.1.

5.5 Options for attk

`pscore(`*scorevar*`)` specifies the name of the user-provided variable containing the estimated propensity score. If this option is not specified, `attk` will estimate the propensity score with the specification provided in *varlist* using a probit.

`logit` requests that a logit estimation of the propensity score be used instead of the default probit model when the option `pscore(`*scorevar*`)` is not specified by the user.

`index` requires the use of the linear index as the propensity score when the option `pscore(`*scorevar*`)` is not specified by the user.

`epan` specifies that the Epanechnikov kernel be used rather than the default Gaussian one.

`bwidth(#)` specifies the bandwidth to be used when choosing the `epan` option. The default is 0.06. This option produces an effect only if the Epanechnikov kernel is requested.

5.6 Options for atts

`pscore(`*scorevar*`)` is a compulsory option that specifies the name of the user-provided variable containing the estimated propensity score.

`blockid(`*blockvar*`)` is a compulsory option and specifies the name of the user-provided variable containing the block identifier of the estimated propensity score.

6 Example: NSW - PSID data

We use data from Dehejia and Wahba (1999), DW for short, which is based on Lalonde's (1986) seminal study of the comparison between *experimental* and *nonexperimental* methods for the evaluation of causal effects. The data combine the treated units from a randomized evaluation of the National Supported Work (NSW) demonstration with nonexperimental comparison units drawn from survey data. For the purpose of this section, we restrict our analysis to the so-called NSW-PSID-1 subsample, consisting of the male NSW treatment units and the largest of the three PSID subsamples (see DW99 for more detail). We use this dataset for two reasons: first, it is widely known in labor economics (starting with Lalonde (1986), re-analyzed by Dehejia and Wahba (1999 and 2002) and by Smith and Todd (2003)) to illustrate the working of propensity score and matching techniques. Second, the data are publicly available at Rajeev Dehejia's web site under the following address: *http://www.columbia.edu/˜rd247/nswdata.html*. We tried to replicate the results produced by Dehejia and Wahba (1999) but—similar to Smith and Todd (2003)—have not been able to numerically replicate all of their estimates because of lack of detailed information in some crucial instances (e.g., number of blocks used in stratification, significance levels, exact procedure for testing the balancing property). However, we get qualitatively similar results. The outcome of interest is RE78 (real earnings in 1978); the treatment T is participation in the NSW treatment group. Control variables are age, education, Black (1 if black, 0 otherwise), Hispanic (1

if Hispanic, 0 otherwise), married (1 if married, 0 otherwise), nodegree (1 if no degree, 0 otherwise), RE75 (earnings in 1975), and RE74 (earnings in 1974). The treatment group contains 185 observations and the control group contains 2,490 observations, so the total number of observations is 2,675.

6.1 Output from pscore

The output from running `pscore` using the DW99 specification is as follows:[6][7]

```
. pscore T age age2 educ educ2 marr black hisp RE74 RE75 RE742 RE752 blackU74,
>  pscore(mypscore) blockid(myblock) comsup numblo(5) level(0.005) logit

**************************************************
Algorithm to estimate the propensity score
**************************************************

The treatment is T
```

T	Freq.	Percent	Cum.
0	2490	93.08	93.08
1	185	6.92	100.00
Total	2675	100.00	

```
Estimation of the propensity score
Iteration 0:   log likelihood = -672.64954
  (output omitted)
Iteration 9:   log likelihood = -204.97537
Logit estimates                                   Number of obs   =       2675
                                                  LR chi2(12)     =     935.35
                                                  Prob > chi2     =     0.0000
Log likelihood = -204.97537                       Pseudo R2       =     0.6953
```

T	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
age	.3316904	.1203295	2.76	0.006	.0958489	.5675318
age2	-.0063668	.0018554	-3.43	0.001	-.0100033	-.0027303
educ	.8492683	.3477041	2.44	0.015	.1677807	1.530756
educ2	-.0506202	.0172492	-2.93	0.003	-.084428	-.0168124
marr	-1.885542	.2993282	-6.30	0.000	-2.472214	-1.298869
black	1.135973	.3517793	3.23	0.001	.446498	1.825447
hisp	1.96902	.5668567	3.47	0.001	.8580017	3.080039
RE74	-.0001059	.0000353	-3.00	0.003	-.000175	-.0000368
RE75	-.0002169	.0000414	-5.24	0.000	-.000298	-.0001357
RE742	2.39e-09	6.43e-10	3.72	0.000	1.13e-09	3.65e-09
RE752	1.36e-10	6.55e-10	0.21	0.836	-1.15e-09	1.42e-09
blackU74	2.144129	.4268089	5.02	0.000	1.307599	2.980659
_cons	-7.474742	2.443502	-3.06	0.002	-12.26392	-2.685566

```
note: 22 failures and 0 successes completely determined.
```

[6] `educ2` denotes squared education, `RE742` and `RE752` denote the square of RE74 and RE75, respectively, and `blackU74` is the interaction of black and a dummy for nonemployment (i.e., zero earnings) in 1974.

[7] Note that when specifying the `detail` option, (even) more detailed output is displayed documenting the steps performed to obtain the final results.

```
Note: the common support option has been selected
The region of common support is [.00061067, .9752541]

Description of the estimated propensity score
in region of common support
                Estimated propensity score
------------------------------------------------------------
      Percentiles      Smallest
 1%     .0006426       .0006107
 5%     .0008025       .0006149
10%     .0010932       .0006159       Obs                1342
25%     .0023546        .000618       Sum of Wgt.        1342
50%     .0106667                      Mean            .1377463
                       Largest        Std. Dev.       .2746627
75%     .0757115        .974804
90%     .6250823       .9749805       Variance        .0754396
95%      .949302       .9752244       Skewness        2.185182
99%      .970598       .9752541       Kurtosis        6.360726

*****************************************************
Step 1: Identification of the optimal number of blocks
Use option detail if you want more detailed output
*****************************************************

The final number of blocks is 7

This number of blocks ensures that the mean propensity score
is not different for treated and controls in each block
```

Following the algorithm described in Section 2, blocks for which the average propensity scores of treated and controls differ are split in half. The algorithm continues until, in all blocks, the average propensity score of treated and controls does not differ. In our case, this happens for a number of seven blocks. Thereafter, `pscore` proceeds to the test of the balancing property for each covariate.

```
*********************************************************
Step 2: Test of balancing property of the propensity score
Use option detail if you want more detailed output
*********************************************************

The balancing property is satisfied
```

When the `detail` option is not specified, the only output produced by `pscore` is a statement saying whether the balancing property is satisfied (which is the case for the DW data with p =0.005) or not. In the latter case, the user is informed for which variable(s) in which block(s) the balancing property failed, and a message is issued suggesting that a different specification of the propensity score be tried.

In case the balancing property holds, the final distribution of treated and controls across blocks is tabulated together with the inferior of each block:

```
This table shows the inferior bound, the number of treated
and the number of controls for each block
```

Inferior of block of pscore	T = 0	T = 1	Total
.0006107	924	7	931
.05	102	4	106
.1	56	7	63
.2	41	28	69
.4	14	21	35
.6	13	20	33
.8	7	98	105
Total	1157	185	1342

```
Note: the common support option has been selected

****************************************
End of the algorithm to estimate the pscore
****************************************
```

Note that we imposed the common support condition in this example using the `comsup` option. Consequently, block identifiers are missing for control observations outside the common support, and the number of observations in the table is 1,342 instead of 2,675.

After running `pscore`, users can proceed to estimate average treatment effects using one of the `att*` programs.

6.2 Output from attnd and attnw

The typical output from `attnd` or `attnw`[8] is

```
. set seed 1221
. attnd RE78 T age age2 educ educ2 marr black hisp RE74 RE75 RE742 RE752 blackU74,
>  comsup boot reps(100) dots logit

 The program is searching the nearest neighbor of each treated unit.
 This operation may take a while.

ATT estimation with Nearest Neighbor Matching method
(random draw version)
Analytical standard errors
```

n. treat.	n. contr.	ATT	Std. Err.	t
185	57	1667.644	2113.592	0.789

```
Note: the numbers of treated and controls refer to actual
nearest neighbour matches
```

[8] Remember that `attnd` and `attnw` will generally give the same results (except for bootstrapped standard errors), unless there are only discrete covariates and multiple nearest neighbors. This is not the case in our example; therefore, to save space, we report here only the output of `attnd`.

```
Bootstrapping of standard errors
command:     attnd RE78 T age age2 educ educ2 marr black hisp RE74 RE75 RE742 R
> E752 blackU74     , pscore() logit  comsup
statistic:   r(attnd)
(obs=2675)
.....................................................................
> ..................
Bootstrap statistics
```

Variable	Reps	Observed	Bias	Std. Err.	[95% Conf.	Interval]	
bs1	100	1667.644	-85.68572	1211.026	-735.2937	4070.582	(N)
					-839.9554	3643.178	(P)
					-394.5013	4064.472	(BC)

N = normal, P = percentile, BC = bias-corrected

```
ATT estimation with Nearest Neighbor Matching method
(random draw version)
Bootstrapped standard errors
```

n. treat.	n. contr.	ATT	Std. Err.	t
185	57	1667.644	1211.026	1.377

```
Note: the numbers of treated and controls refer to actual
nearest neighbour matches
```

Note that in this example, only 57 different controls have been matched to the 185 treated. These results are very close to the ones obtained by Dehejia and Wahba (1999).

6.3 Output from attr

For `attr` with radius $r = 0.0001$ we obtain

```
. attr RE78 T age age2 educ educ2 marr black hisp RE74 RE75 RE742 RE752 blackU74,
>  comsup boot reps(100) dots logit radius(0.0001)

 The program is searching for matches of treated units within radius.
 This operation may take a while.

ATT estimation with the Radius Matching method
Analytical standard errors
```

n. treat.	n. contr.	ATT	Std. Err.	t
23	66	-5546.140	2388.723	-2.322

```
Note: the numbers of treated and controls refer to actual
matches within radius
```

```
Bootstrapping of standard errors
command:     attr RE78 T age age2 educ educ2 marr black hisp RE74 RE75 RE742 RE
> 752 blackU74     , pscore() logit  comsup radius(.0001)
statistic:   r(attr)
(obs=2675)
.....................................................................
> .....................
Bootstrap statistics
```

Variable	Reps	Observed	Bias	Std. Err.	[95% Conf.	Interval]	
bs1	100	-5546.14	425.988	4657.081	-14786.8	3694.519	(N)
					-13867.87	5668.455	(P)
					-13867.87	5668.455	(BC)

N = normal, P = percentile, BC = bias-corrected

```
ATT estimation with the Radius Matching method
Bootstrapped standard errors
```

n. treat.	n. contr.	ATT	Std. Err.	t
23	66	-5546.140	4657.081	-1.191

```
Note: the numbers of treated and controls refer to actual
matches within radius
```

The large difference with respect to the caliper matching results of Dehejia and Wahba (2002) comes from the fact that caliper matching differs from radius matching in that the nearest control is used as a match if a treated unit has no control units within radius r. Whereas caliper matching uses all treated units, our method only uses those treated that have control matches within radius r (here, 23 out of 185 treated). This example illustrates the sensitivity of the results to extreme assumptions used in the matching procedure. If the radius is chosen to be very small, many treated units are not matched and the results are no longer representative of the population of treated. For a more detailed discussion of this issue, see Smith and Todd (2003).

6.4 Output from attk

For `attk`, the results are as follows:[9]

```
. attk RE78 T age age2 educ educ2 marr black hisp RE74 RE75 RE742 RE752 blackU74,
>  comsup boot reps(100) dots logit

 The program is searching for matches of each treated unit.
 This operation may take a while.
```

[9]Note that Dehejia and Wahba do not present results for kernel matching.

```
ATT estimation with the Kernel Matching method
```

n. treat.	n. contr.	ATT	Std. Err.	t
185	1157	1537.943	.	.

```
Note: Analytical standard errors cannot be computed. Use
the bootstrap option to get bootstrapped standard errors.

Bootstrapping of standard errors
command:     attk RE78 T age age2 educ educ2 marr black hisp RE74 RE75 RE742 RE
> 752 blackU74     , pscore() comsup logit   bwidth(.06)
statistic:   r(attk)
(obs=2675)
.....................................................................
> .....................
Bootstrap statistics
```

Variable	Reps	Observed	Bias	Std. Err.	[95% Conf.	Interval]	
bs1	100	1537.943	-51.50918	1016.874	-479.755	3555.642	(N)
					-439.9654	3601.629	(P)
					-343.8961	3826.322	(BC)

```
                              N = normal, P = percentile, BC = bias-corrected

ATT estimation with the Kernel Matching method
Bootstrapped standard errors
```

n. treat.	n. contr.	ATT	Std. Err.	t
185	1157	1537.943	1016.874	1.512

In kernel matching, all treated, as well as all controls (in the common support which has been imposed here), are used. The estimate of the ATT is quite close to the one obtained with nearest-neighbor matching.

6.5 Output from atts

Finally, for `atts` with the blocks obtained in pscore:[10]

[10]Note that there are two special cases as concerns the computation of the ATT and its analytical standard error. First, if there is no treated and/or no control unit in one (or more) of the blocks, the ATT is computed on the remaining blocks that practically amounts to imposing a (block-based) common support condition. Second, if there is exactly one treated and/or one control in one (or more) of the blocks, the ATT in that block can still be computed but the standard error cannot. In this case, `atts` will produce missing values for the standard error. However, bootstrapped standard errors can still be computed.

```
. atts RE78 T, pscore(mypscore) blockid(myblock) comsup boot reps(100) dots

ATT estimation with the Stratification method
Analytical standard errors

------------------------------------------------------------------------------
n. treat.   n. contr.         ATT   Std. Err.           t
------------------------------------------------------------------------------
      185        1157    2208.600     777.866       2.839
------------------------------------------------------------------------------

Bootstrapping of standard errors
command:     atts RE78 T  , pscore(mypscore) blockid(myblock) comsup
statistic:   r(atts)
(obs=2675)
..................................................................................
> ..................
Bootstrap statistics
Variable |    Reps   Observed       Bias   Std. Err.   [95% Conf. Interval]
---------+--------------------------------------------------------------------
     bs1 |     100     2208.6    96.6845   850.9957    520.0395   3897.16  (N)
         |                                             570.5178  4012.478  (P)
         |                                             778.2358  4184.918 (BC)
------------------------------------------------------------------------------
                                N = normal, P = percentile, BC = bias-corrected

ATT estimation with the Stratification method
Bootstrapped standard errors

------------------------------------------------------------------------------
n. treat.   n. contr.         ATT   Std. Err.           t
------------------------------------------------------------------------------
      185        1157    2208.600     850.996       2.595
------------------------------------------------------------------------------
```

Here, the difference with respect to the DW99 results is slightly bigger than for nearest-neighbor matching. This might be explained by a different number of blocks used in stratification, different significance levels, or a different procedure for testing the balancing property (see general remark at the beginning of the section 6). But overall, the results obtained by `attnw`, `attk`, and `atts` are quite close to each other, and taken together give evidence of a positive ATT in the range of 1500–2200 associated with the NSW demonstration (when evaluated with nonexperimental comparison groups), which is close to the experimental estimates of about 1700.

(*Continued on next page*)

7 Saved Results

The `att*` commands save in `r()`:

Scalars

`r(nt*)`	number of treated used in the computation of `att*`
`r(nc*)`	number of controls used in the computation of `att*`
`r(att*)`	ATT obtained by `att*`
`r(seatt*)`	analytical standard error for `att*`
`r(tsatt*)`	analytical t statistic for `att*`
`r(bseatt*)`	bootstrapped standard error for `att*`
`r(btsatt*)`	bootstrapped t statistic for `att*`
`r(mean1)`	mean outcome of matched treated for `attk`
`r(mean0)`	mean outcome of matched controls for `attk`

8 Acknowledgments

We thank Rajeev Dehejia for providing us with his data. We would like to thank Marco Caliendo, Rajeev Dehejia, Hendrik Juerges, Torsten Persson, Anna Sanz de Galdeano, Barbara Sianesi and Daniela Vuri for very helpful discussions and for testing our programs on their data. The comments of an anonymous referee helped to considerably improve the paper and the programs.

9 References

Dehejia, R. H. and S. Wahba. 1999. Causal effects in nonexperimental studies: Reevaluation of the evaluation of training programs. *Journal of the American Statistical Association* 94: 1043–1062.

—. 2002. Propensity score matching methods for non-experimental causal studies. *Review of Economics and Stiatistics* 84(1): 151–161.

Imbens, G. W. 2000. The role of propensity score in estimating dose–response functions. *Biometrika* 87(3): 706–710.

Lalonde, R. 1986. Evaluating the econometric evaluations of training programs. *American Economic Review* 76: 604–620.

Lechner, M. 2001. A note on the common support problem in applied evaluation studies. Discussion Paper 2001–02. Department of Economics, University of St. Gallen.

Rosenbaum, P. R. and D. B. Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70(1): 41–55.

Smith, J. and P. Todd. 2003. Does matching overcome Lalonde's critique of non experimental estimators? *Journal of Econometrics*. Forthcoming.

About the Authors

Sascha O. Becker is assistant professor of Economics at the University of Munich, Germany. He is also affiliated with CESifo and IZA.

Andrea Ichino is professor of Economics at the European University Institute, Florence, Italy. He is also affiliated with CESifo, CEPR and IZA.

Revised and improved versions of the programs may become available in the future on our web pages (*http://www.sobecker.de* and *http://www.iue.it/Personal/Ichino*).

Please address any correspondence to Sascha O. Becker (so.b@gmx.net) or Andrea Ichino (andrea.ichino@iue.it).