Aspects of Causal Inference within the Evenly Matchable Population: The Average Treatment Effect on the Evenly Matchable Units, Visually Guided Cohort Selection, and Bagged One-to-One Matching

By

Lauren Ruth Samuels

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Biostatistics

May, 2017

Nashville, Tennessee

Approved:

Bryan E. Shepherd, Ph.D.

Robert A. Greevy, Jr., Ph.D.

Matthew S. Shotwell, Ph.D.

Meira Epplein, Ph.D.

ACKNOWLEDGEMENTS

I feel fortunate to have received a great deal of support and encouragement in preparing this dissertation.

Although this work was not funded directly by any outside source, I am grateful to Dr. Angela Jefferson and the Vanderbilt Memory and Alzheimer's Center for supporting me financially as a research assistant for the last four years and to Dr. Dandan Liu for her supervision of my work in that collaboration.

All four members of my dissertation committee— Dr. Meira Epplein, Dr. Bryan Shepherd, Dr. Matthew Shotwell, and my advisor, Dr. Robert Greevy— have provided valuable and caring guidance and have pushed my research and writing forward with their thought-provoking questions. It has been a privilege to work with all of them. I would especially like to thank Dr. Greevy for his wisdom, patience, and kindness both during the dissertation process and in the years of coursework and exams leading up to it.

I am blessed to have a loving and supportive family, whose belief in me gave me the courage to apply to graduate school and has sustained me throughout the program. I close with deep thanks to my parents, my sister, and my life partner, Lisa.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ATC Average treatment effect on the control units

ATE Average treatment effect over all units

ATM Average treatment effect on the evenly matchable units

ATT Average treatment effect on the treated units

BOOM Bagged one-to-one matching

IPW Inverse probability weighting

MSE Mean squared error

MW Matching weight

OLS Ordinary least squares

OOM One-to-one matching

PS Propensity score

SE Standard error

SMD Standardized mean difference

Chapter 1

Introduction

Observational studies are often concerned with estimating the average effect of a treatment or exposure on all or some of the units in a sample or population. In many cases, the average treatment effect on the treated units within a sample or population (ATT) is of particular interest. Using the potential-outcomes framework and notation from Ho et al. (2007), which is based on the work of Neyman et al. (1935), Rubin (1974), and others, we have

$$\text{ATT} \equiv \mathsf{E}[Y_i(1) - Y_i(0) \mid T_i = 1], \tag{1.1}$$

where $Y_i(1)$ is the outcome that unit i would experience under the treatment or exposure of interest, $Y_i(0)$ is the outcome the unit would experience under the control condition, and T_i is the binary treatment indicator. Although the ATT is an intuitively appealing target of inference, Rosenbaum (2012) and Li and Greene (2013) argue for the consideration of a slightly different treatment effect: the effect on what Rosenbaum calls the "marginal" units, those units that could have conceivably received either treatment. In this dissertation I explore aspects of this alternative estimand in three papers, first providing a formal definition and then providing tools related to its estimation.

Building on the ideas in Rosenbaum (2012) and Li and Greene (2013), Chapter 2 first defines the evenly matchable units in a sample or population and then defines the average treatment effect on the evenly matchable units (ATM). The chapter continues with an examination of currently available methods that can be used to estimate the ATM and then introduces new estimation approaches, including bagged one-to-one matching (BOOM), which is described in detail in Chapter 4, and three new weighting techniques. Chapter 2 then illustrates several of these ATM estimation techniques in a case study. The chapter closes with a discussion of the value of the ATM as an estimand in causal inference and of the advantages and disadvantages of the various ATM estimation techniques.

A crucial first step in the estimation of the ATM or of any other treatment effect is the examination of the covariate balance in the available sample and, if necessary, the selection of an appropriate cohort in which to study the treatment of interest—that is, "pruning" the original dataset (Ho et al., 2007). Chapter 3 describes Visual

Pruner, a new web application designed to facilitate this task. The freely available app allows analysts to use information from both baseline covariate distributions and estimated propensity scores to create transparent, covariate-based study inclusion criteria. Although the app is designed to be used in studies regardless of the eventual target of inference, Visual Pruner can be especially helpful when estimation of the ATM is of interest, because the app allows the examination of standardized mean differences in a weighted cohort created using the matching weights developed by Li and Greene (2013), one of the ATM estimation techniques discussed in Chapter 2.

Once researchers have examined the initial covariate balance in a sample and pruned the sample as necessary, a variety of ATM estimation techniques are available, as discussed in Chapter 2. Chapter 4 presents a detailed introduction to bagged one-to-one matching (BOOM), which can be used to estimate either the ATM or the ATT, depending on the options chosen and the sample at hand. BOOM combines the bias-reducing properties of one-to-one matching with the variance-reducing properties of bootstrap aggregating, or bagging (Breiman, 1996). After delineating the BOOM algorithm, Chapter 4 presents a simulation study exploring the BOOM estimator's performance under different types of model misspecification and in comparison to established estimation techniques. The chapter closes with a brief case study in which we take a closer look at the BOOM process in a single dataset.

Taken together, these three papers provide an overview of the ATM as an estimand and several valuable tools related to its estimation. While the ATM is the thread that ties the three papers together, both Visual Pruner and BOOM can be used when estimands other than the ATM are of interest, and even in cases where the researcher is not explicitly interested in estimating a causal effect. It is my hope that consideration of the ATM as a target of inference will be useful to other researchers working with observational studies, and that the Visual Pruner app and the BOOM estimation and weighting processes will be valuable additions to the design and analysis tools in this challenging field.

Chapter 2

The Average Treatment Effect on the Evenly Matchable Units (ATM): A Valuable Estimand in Causal Inference

2.1 Introduction

Observational studies are often concerned with estimating the average effect of a treatment or exposure on a sample or population. In many cases, the average treatment effect on the treated units within that sample or population (ATT) is of particular interest. Using the potential-outcomes framework and notation from Ho et al. (2007), which is based on the work of Jerzy Neyman (Neyman et al., 1935), Donald Rubin (Rubin, 1974), and others, we have

$$\text{ATT} \equiv \mathsf{E}[Y_i(1) - Y_i(0) \mid T_i = 1], \tag{2.1}$$

where $Y_i(1)$ is the outcome that unit i would experience under the treatment or exposure of interest, $Y_i(0)$ is the outcome the unit would experience under the control condition, and T_i is the binary treatment indicator. In theory, the ATT can be estimated within a weighted or matched cohort created using propensity scores. In practice, though, the quantity estimated by weighting-based estimators can be biased away from the ATT by the truncation of weights in an attempt to control the variance of the estimator (Austin and Stuart, 2015; Cole and Hernán, 2008); and, as Li and Greene (2013) point out, the estimand of one commonly used propensity-score matching-based estimator is not in general the ATT. Thus, while the ATT may be of interest, many studies that attempt to estimate the ATT are in fact providing either biased estimates of the ATT or (perhaps unbiased) estimates of another quantity altogether.

Li and Greene (2013) argue that the quantity that *is* estimated by the above-mentioned commonly used propensity-score matching-based estimator is an important and distinct quantity in causal inference, and they also introduce a new weighting scheme that can be used to estimate this quantity. In this paper we extend their work by naming and formally defining this estimand while also broadening its scope to encompass studies where the propensity score is not of interest. We propose to call the new estimand the average treatment effect on the evenly matchable units, or ATM. In Section 2.2 we formally define the ATM; in Section 2.3 we introduce additional ways of estimating it; and in Section 2.4 we illustrate its use with a case

study. The paper concludes with a discussion (Section 2.5).

2.2 Defining the ATM

The popular estimator discussed by Li and Greene (2013) is the difference in group means in a cohort selected through a matching algorithm referred to as "pair matching" in that paper and "PS caliper pair matching" here: one-to-one greedy (nearest-neighbor) matching on the propensity score without replacement and with the use of a caliper, a constraint on the maximum allowable distance within a matched pair.[1] Before providing a formal definition of the ATM, we provide an illustration to complement Li and Greene's work. Figure 2.1 shows overlapping histograms of the estimated propensity scores from three hypothetical studies in which a researcher might want to conduct PS caliper pair matching. Although the matching would in reality be conducted within a caliper around each treated unit's propensity score, for the sake of illustration we will proceed as though matching will be conducted within each histogram bin.

In the top plot of Figure 2.1, the control units outnumber the treated units across the distribution of propensity scores, such that in every bin of the histogram, each treated unit can be matched to a control unit. Because all of the treated units will be included in the matched cohort, the cohort selected by PS caliper pair matching can be used to estimate the ATT. In the middle panel, we see that in the lower part of the range of propensity scores, all of the treated units can be matched, while in the higher part of the range, none of the treated units can be matched. This is a case where, although the PS caliper pair matching estimator cannot estimate the ATT on the full set of treated units, it can estimate the ATT on a portion of the treated units, a quantity that is sometimes called the local ATT (Iacus et al., 2012).

In the bottom plot, however, we see that all of the treated units in the approximate range (0, 0.5) can be matched, but that in the upper part of the propensity score range, only some of the treated units in each bin can be matched. Under PS caliper pair matching, any particular matched cohort would contain only some of the treated units, with the included units depending on the order of matching. Li and Greene (2013) demonstrate that in cases like this, where the propensity score distribution contains regions in which some but not all of the treated units can be matched, the PS caliper pair matching estimator does not estimate the ATT. Li and Greene argue that the estimand of the PS caliper pair matching estimator is nevertheless a useful

[1] Li and Greene (2013) do not specify an exact caliper, but they do specify that the caliper should grow smaller with larger sample sizes.

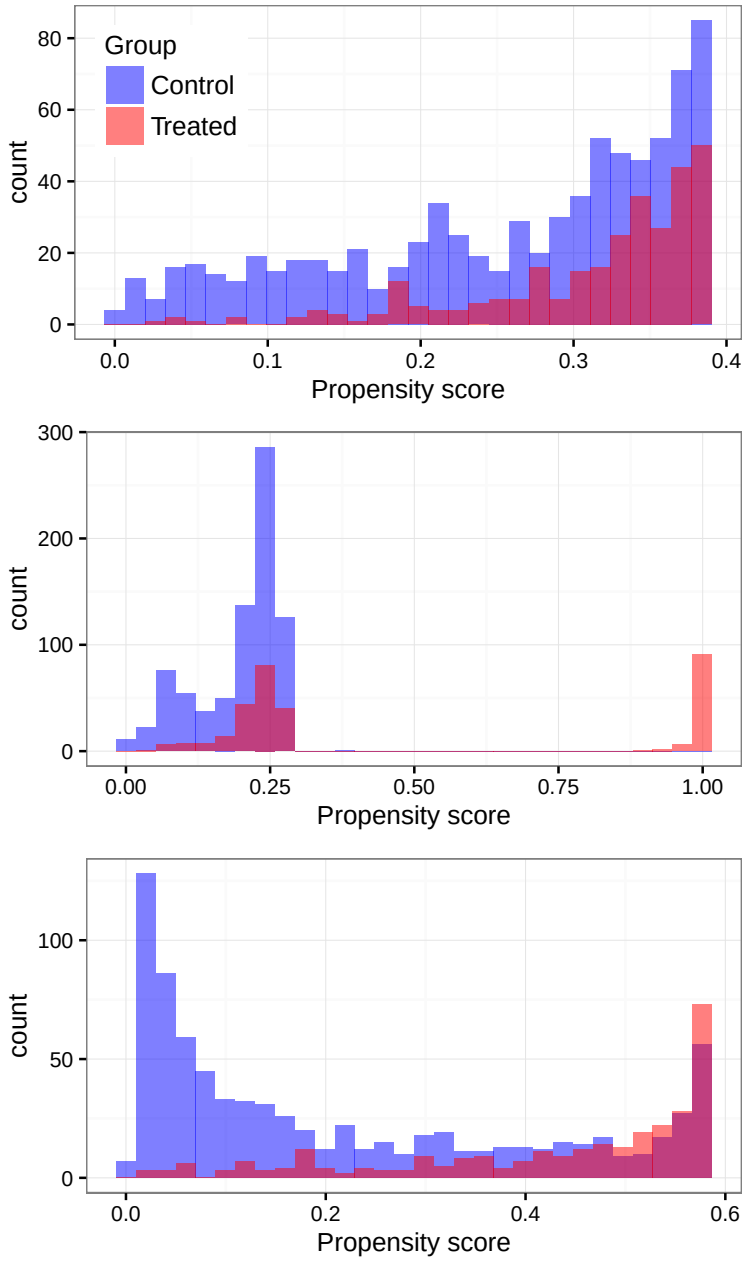

Figure 2.1: Propensity score histograms from three hypothetical studies. In each panel, the two histograms are overlaid (not stacked). In the top panel, the control group completely overlaps the treatment group. In the middle panel, the control group completely fails to overlap the treatment group in one region, while the bottom panel shows partial lack of overlap.

quantity, and in many cases a more useful quantity than the ATT. We propose calling this estimand the average treatment effect on the evenly matchable units (ATM), and we clarify what we mean by "evenly matchable" in the following paragraph.

Within each bin of the histograms in Figure 2.1, we define the set of "evenly matchable units" as those units that belong to the least prevalent exposure. If a bin contains equal numbers of treated and control units, we say that all the units in that bin are evenly matchable. Overall for each histogram, the set of evenly matchable units consists of the union of the sets of evenly matchable units from each bin. Equivalently, for the histograms in Figure 2.1, we can say that a unit is evenly matchable if its bin contains at least as many units from the opposite group as from its own group. In the bottom panel, within the bins for propensity scores below about 0.5, the treated units are evenly matchable, and within the bins in the upper part of the range, the control units are evenly matchable. Thus in this case, the set of evenly matched units consists of the treated units that have propensity scores lower than about 0.5 and the control units that have propensity scores higher than that value.

More broadly, for any sample or population, within each localized region of the covariate space, we define the set of evenly matchable units as those units that belong to the least prevalent exposure. If the localized region contains equal numbers of treated and control units, we say that all the units in that region are evenly matchable. If we "bin" the covariate space into nonoverlapping localized regions with no gaps, the complete set of evenly matchable units is the union of the sets of evenly matchable units over the whole covariate space. We can also say that a unit is evenly matchable if the localized region of covariate space centered on that unit contains at least as many units from the opposite group as from its own group. These two definitions become equivalent as the sample size approaches infinity and the localized regions become arbitrarily small. Here we use "covariate space" broadly to refer to either the original covariate space of the study or a (possibly dimension-reducing) transformation thereof. Commonly used dimension-reducing transformations include the propensity score, logit propensity score, and prognostic score (Hansen, 2008). Other transformations, such as the centering, scaling and sphering that yield the Mahalanobis distance, are also possible. Because the evenly matchable units in terms of one transformation are not necessarily the same as the evenly matchable units in terms of another transformation, we index the ATM by the type of transformation, e.g. ATM_{PS} for propensity scores, ATM_{Prog} for prognostic scores, etc. Thus, using

the notation introduced above in Equation 2.1, we have

$$\text{ATM}_d \equiv \mathsf{E}[Y_i(1) - Y_i(0) \mid M_{di} = 1], \tag{2.2}$$

where d denotes the distance measure or other transformation used and $M_{di} = 1$ if unit i is in the set of evenly matchable units under that distance measure, 0 otherwise. As with the ATT and other estimands in causal inference, we can speak of the ATM for a sample (SATM) or for the population from which a sample is drawn (PATM).

2.3 Techniques for estimating the ATM

Although we have defined the ATM in terms of the evenly matchable units, explicit identification of the evenly matchable units is not necessary for estimating the ATM. In fact, most techniques for estimating the ATM bypass the explicit identification of the evenly matchable units. In this section we present a variety of methods that can be used to estimate the ATM. We review the two approaches discussed in Li and Greene (2013) and introduce several other techniques. All of these techniques can either be combined with or followed by further covariate adjustment (e.g., through regression), as suggested in Ho et al. (2007). For simplicity, however, we present only their basic forms here.

2.3.1 Variations on "caliper pair matching"

As noted above, ATM_{PS} can be estimated using the cohort selected by PS caliper pair matching. Similarly, $\text{ATM}_{\text{LogitPS}}$, ATM_{Prog}, and ATM_{MD} can be estimated using caliper pair matching on the logit propensity score, the prognostic score, or the Mahalanobis distance, respectively. The algorithm we are calling "caliper pair matching"— that is, greedy one-to-one matching without replacement and with the use of a caliper— has four modifiable aspects. We now consider modifying each of these aspects one at a time to explore which (if any) of them are essential in selecting a cohort that can be used to estimate the ATM:

Greedy matching Using optimal, rather than greedy, one-to-one matching with a caliper (or its equivalent) can produce a cohort that can be used to estimate the ATM. (Optimal one-to-one matching is always conducted without replacement.) While the method of selecting pairs differs from that of PS caliper pair matching, because of the use of the caliper, the final cohort will contain all of the treated units from the regions where the treated units are evenly matchable and all of

the control units from the regions where the control units are evenly matchable and will thus be suitable for estimation of the ATM.

One-to-one matching Maintaining the use of matching without replacement and with a caliper, using 1:k matching with fixed k rather than 1:1 matching can select a cohort from which the ATM can be estimated only when there are at least k times as many controls as treated units within every localized region of the covariate space, in which case the ATM is equivalent to the ATT. Similarly, 1:k matching with variable k can be used to estimate the ATM only when the ATM is equivalent to the ATT; in this case, the number of controls must be at least equal to the number of treated units within every localized region of the covariate space. Note that the variable-k variation of the PS caliper pair matching algorithm must be followed by weighting and/or further covariate adjustment; it does not produce a cohort that can be used directly in estimation.

Matching without replacement For any matching ratio and with or without the use of a caliper, matching with replacement rather than without replacement selects treated units into the matched cohort without regard to whether they are in a region where the treated units are evenly matchable. Furthermore, it does not guarantee the inclusion of all of the control units from the regions where the control units are evenly matchable (because another control unit might be a slightly better match). Thus, matching with replacement cannot in general be used for estimation of the ATM.

Matching with a caliper Any form of matching without a caliper does not allow treated units to be excluded from the matched cohort, so matching without a caliper does not estimate the ATM unless there are ample controls near every treated unit, in which case the ATM is equivalent to the ATT. (Note that while exact matching does not explicitly specify a caliper, it essentially uses a caliper of zero; thus exact matching can be used to estimate the ATM in an appropriately dense covariate space.)

In summary, then, while many of the variations on caliper pair matching discussed above are suitable for estimating both the ATM and the ATT when the ATT is a viable estimand, they are not in general suitable for estimating the ATM. Of the elements of PS caliper pair matching discussed above, the only one that can be safely varied if ATM estimation is of interest is the use of greedy matching. Both greedy one-to-one matching without replacement and with a caliper and optimal one-to-one

matching with a caliper allow the exclusion of treated units in areas of the covariate space where treated units are not evenly matchable and guarantee the inclusion of all of the control units from the regions where the control units are evenly matchable. That is, for each group, both algorithms include all units from the areas where that group is evenly matchable, and they allow the exclusion of units from that group in areas where that group is not evenly matchable. Any other matching algorithm that has these two features and that produces a cohort that does not require the application of weights before estimation can also be used in estimation of the ATM.

2.3.2 Bagged one-to-one matching (BOOM)

Bagged one-to-one matching (BOOM), introduced in a forthcoming manuscript (Samuels and Greevy, Jr., 2016*a*) and in Chapter 4, conducts one-to-one matching in many bootstrap resamples of the original sample in order to obtain a matching-based estimator with reduced variance. The BOOM process can be used with any variation of one-to-one matching, and the BOOM estimator will have the same estimand as the repeated one-to-one matching procedure. Thus, BOOM using either greedy matching without replacement or optimal matching will estimate the ATM as long as an appropriate caliper is used.

2.3.3 Matching weights (ATM_{PS} only)

The main focus of Li and Greene (2013) is to introduce a weighting-based estimator that asymptotically has the same estimand as the PS caliper pair matching estimator (or, in the terminology of the present paper, that estimates ATM_{PS}). Because they rely on propensity scores, Li and Greene's weights, called "matching weights," can be used to estimate the ATM only when the propensity score is the primary distance measure or transformation of interest. Li and Greene define the matching weight for unit i as

$$MW_i \equiv \frac{\min(1 - e_i, e_i)}{T_i e_i + (1 - T_i)(1 - e_i)}, \tag{2.3}$$

where e_i is the propensity score for unit i and T_i is that unit's treatment indicator (notation changed slightly from original paper). Li and Greene explain that the matching weight can be thought of as the probability of being included in a matched cohort selected through PS caliper pair matching, and they note that the matching weights can be used as sampling weights in a similar fashion to inverse probability weights, both in estimation without further covariate adjustment and in doubly robust

estimation.

As a complement to the exposition of matching weights in Li and Greene (2013), in Figure 2.2 we show, across the range of possible propensity scores in any study, matching weights (bottom right) in comparison to standard inverse probability weights for estimating three common estimands in causal inference: the ATT (top left), the ATC (average effect of treatment on the control units, top right), and the ATE (average treatment effect over all units, bottom left) (Austin, 2011). In the top left panel we see that the ATT weighting system gives each treated unit a weight of one, and then downweights the control units with propensity scores less than 0.5 and upweights the control units with propensity scores greater than 0.5; the effect will be to create a synthetic control group that has the same distribution of propensity scores (and, asymptotically, the same marginal distribution of covariates) as the treated group. The ATC weights (top right) are mirror opposites of the ATT weights, with the groups exchanged. Looking at the matching weights (bottom right), we see that for propensity scores below 0.5, the weights resemble the ATT weights, and for propensity scores above 0.5, the weights resemble the ATC weights. In fact, we can show that the matching weight for each unit is exactly equal to the minimum of the standard inverse probability weight used for estimation of the ATC and the standard inverse probability weight used for estimation of the ATT:

$$\begin{aligned} MW_i &\equiv \frac{\min(1-e_i, e_i)}{T_i e_i + (1-T_i)(1-e_i)} \\ &= \min\left(\frac{(1-e_i)}{T_i e_i + (1-T_i)(1-e_i)}, \frac{e_i}{T_i e_i + (1-T_i)(1-e_i)}\right) \\ &= \min(ATCW_i,\ ATTW_i), \end{aligned}$$

where $ATCW_i$ and $ATTW_i$ are the standard ATC and ATT weights, respectively, for unit i. In contrast, the ATE weighting system (bottom left) weights the units in each group so that, asymptotically, the covariate distribution in each group will resemble the distribution in the original sample as a whole. Looking at the ATT, ATC, and ATE weighting systems, we see the possibility for extremely large weights that can inflate the variances of these estimators. With matching weights, however, the largest possible weight is 1, so there is reduced potential for variance inflation.

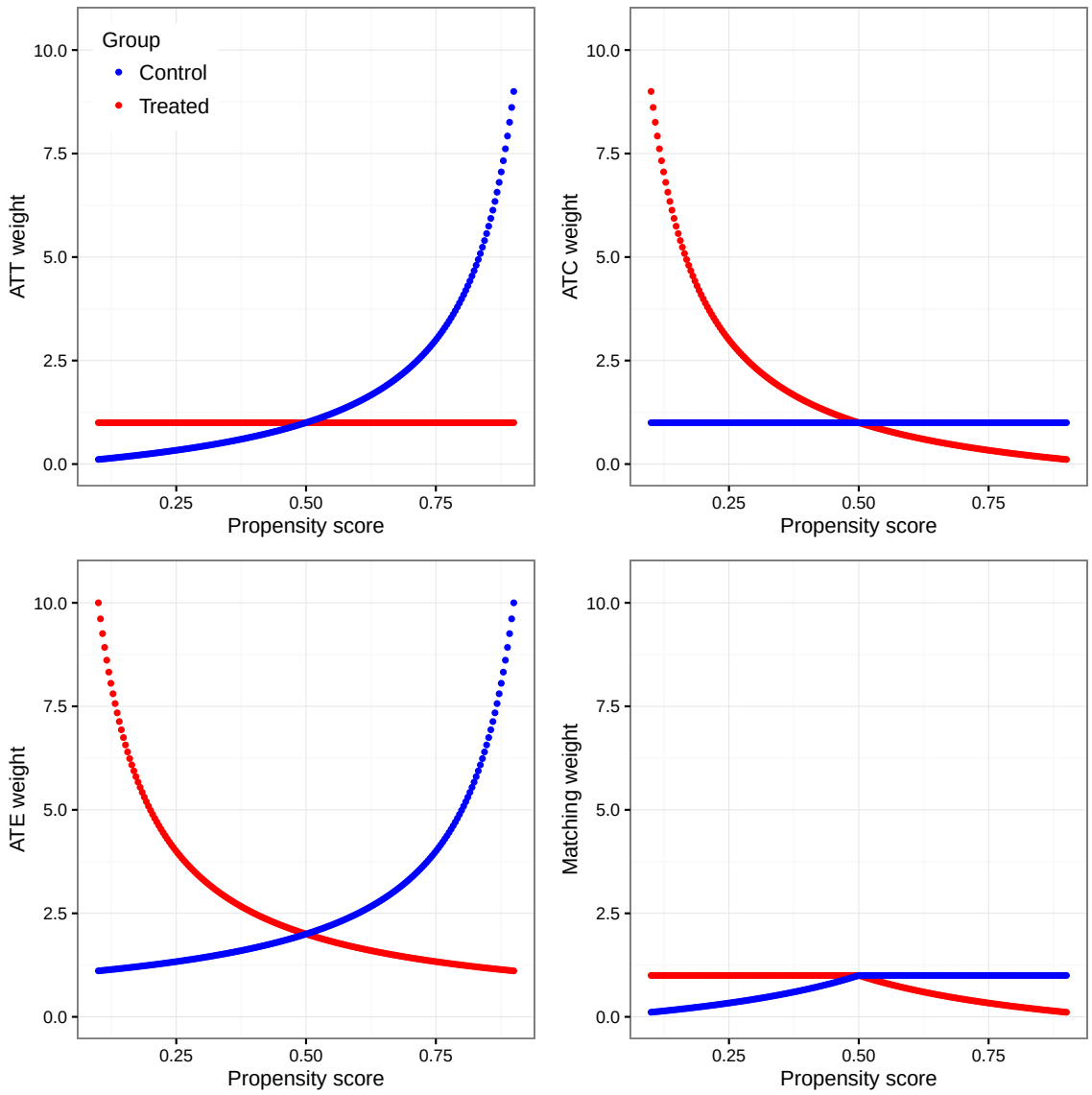

Figure 2.2: A comparison of weighting schemes. The matching weights developed by Li and Greene (2013) (bottom right) can be used to estimate the ATM; weights for selected other estimands are shown in the other three panels. ATT: average treatment effect on the treated units; ATC: average treatment effect on the control units; ATE: average treatment effect over all units.

2.3.4 Three new weighting-based approaches

Li and Greene (2013) argue that estimation using matching weights is preferable to estimation using PS caliper pair matching because the weighting framework is more amenable to both accurate variance estimation and formal double robust estimation. Furthermore, in simulations they found that the matching weight (MW) estimator was more efficient than the PS caliper pair matching estimator. Related to this advantage is the MW estimator's use of at least a small amount of information from each unit, a feature that many researchers find appealing. Thus, when the ATM is the estimand of interest and the propensity score is of interest as a measure of distance, researchers might prefer to use Li and Greene's matching weights rather than PS caliper pair matching. In some studies, however, a different distance measure or transformation of the covariate space is of interest, and the closed-form matching weights introduced by Li and Greene cannot be used directly with these other transformations. In these cases it might be advantageous to have a way to use weights instead of a matching algorithm. Here we introduce ways to obtain weights for ATM estimation when the distance measure of interest is something other than the propensity score.

2.3.4.1 BOOM weights

As mentioned briefly in Samuels and Greevy, Jr. (2016*a*) and in Chapter 4, the BOOM process (Section 2.3.2) automatically generates a weight for each unit. For each unit, the BOOM weight is the number of times that unit is included in a matched cohort, divided by the number of times that unit is included in a bootstrap sample, or

$$BW_i = \frac{\#\text{ times unit } i \text{ is matched}}{\#\text{ times unit } i \text{ is sampled}}$$

(Units that are never included in a bootstrap sample are given a weight of zero; however, if any units are never included in a bootstrap sample, it is likely that the number of resamples is too low.) As noted above, Li and Greene conceive of matching weights as the probability of being included in a matched sample under PS caliper pair matching. Similarly, BOOM weights give the empirical proportion of times a unit is included in a matched sample out of all of its opportunities for inclusion. The BOOM process, then, is a useful way to convert any distance measure or transformation of the covariate space— or any one-to-one matching process— into weights similar in nature to Li and Greene's closed-form matching weights.

A natural question about the BOOM weights is whether a weighted analysis using BOOM weights would give the same point estimate and confidence interval as the

analysis that used BOOM estimation directly. For noncollapsible outcomes or analyses in which further covariate adjustment is done after the matching or weighting, we would not expect the point estimates from the two approaches to be identical. In the case of a collapsible outcome and no direct covariate adjustment, the point estimates from the two methods may differ slightly because the BOOM weights are slightly different from the effective weights of the BOOM estimator. The BOOM weight's denominator uses the actual number of times a unit is sampled to adjust for chance over- or under-sampling of a given unit, while the BOOM estimator effectively uses a denominator of B, the number of bootstrap resamples. Thus the BOOM weights may count a given unit slightly more or less than the BOOM estimator. The difference between the two becomes trivial as B becomes large. For variance estimation, BOOM estimation uses an infinitesimal-jackknife approach, while sampling-weighted analyses generally use robust standard error estimation or Taylor series linearization (Lumley, 2011; Binder, 1983), so we would not necessarily expect confidence intervals to be identical for the two methods. Preliminary work suggests that in practice, however, the confidence intervals for the two methods are extremely close, particularly when the bias-corrected version of the BOOM standard error estimate is used.

Although we have presented BOOM weights as an option for researchers who want to conduct ATM weighting using a distance measure or transformation other than the propensity score, BOOM weights are also an option for researchers who are interested specifically in ATM_{PS}. The BOOM weighting system produces weights that differ slightly from Li and Greene's matching weights, and researchers might prefer one system over the other depending on the details of the study. In any sample, the matching weights will give a nonzero weight to each unit in the sample, but the BOOM weights do not have this constraint. That is, the matching weights rely more heavily on the theoretical distribution of the propensity scores in the population, while the BOOM weights rely more heavily on the empirical distribution of the propensity scores in the sample. This is relevant in situations like the one illustrated in the middle panel of Figure 2.1, where one region of the propensity score distribution contains only treated units. Those treated units would almost certainly have BOOM weight of zero, but they would contribute to the estimate under the matching weights. Ideally, a researcher with a sample like this would prune the dataset first to eliminate the region of complete nonoverlap (Ho et al., 2007). Compared to matching weights, then, BOOM weights are more forgiving of a failure to prune first and are thus more robust to violations of the positivity assumption. BOOM weights also allow more flexibility in the estimation of propensity scores; the standard BOOM algorithm involves

re-estimating the propensity score model within each bootstrap sample, with other variations possible. Another difference between the two weighting systems is that BOOM weights are naturally smoother in the region around 0.5, whereas matching weights exhibit a sharp change at that point. A final consideration, particularly for large datasets, is time: estimation of matching weights is much faster than estimation of BOOM weights.

2.3.4.2 Vicinity weights

Because BOOM is computationally intensive, generating BOOM weights may not be feasible for large data sets. For studies involving large data sets where a distance measure other than the propensity score is of interest, then, a faster way to generate ATM weights using just the raw or transformed data and a caliper would be especially helpful. Here we introduce vicinity weights, whose calculation requires only those two elements, or alternatively a distance matrix and a caliper.

To create vicinity weights in a sample of size N, we begin with either the raw or transformed data or a distance matrix. If we are starting with the data, we first create an $N \times N$ distance matrix where cell i, j gives the distance from unit i to unit j in terms of the transformation of interest (propensity score, prognostic score, Mahalanobis distance, etc.). We then specify a caliper that indicates a "reasonably close" distance under this transformation and with a sample of this size; this could be the same caliper that one would choose for use with a matching algorithm. Then, for each unit i, we identify its vicinity as the set of units (including unit i) whose distance from unit i is less than or equal to the caliper. We calculate the vicinity-based propensity score e^* for unit i as the proportion of the units in its vicinity (including itself) that are in the treated group. Then, just as propensity scores are used to derive matching weights (Equation 2.3), we can use the vicinity-based propensity scores to calculate a vicinity weight for each unit:

$$VW_i \equiv \frac{\min(1 - e_i^*, e_i^*)}{T_i e_i^* + (1 - T_i)(1 - e_i^*)}. \tag{2.4}$$

Because the vicinity-based propensity scores are discrete quantities, the vicinity weights derived from them may exhibit an undesirable degree of discreteness. One approach to lessening this degree of discreteness is to increase the caliper used in calculating the vicinity-based propensity scores, if reasonable. Another approach is to smooth the vicinity-based propensity scores before calculating the vicinity weights. Many smoothing approaches are possible; here we present just one. After calculating

the vicinity-based propensity scores for all N units, we can calculate the smoothed vicinity-based propensity scores as follows: For each unit i, assign to each unit in its vicinity a weight that is a function of its distance from unit i. In particular, we use a Gaussian radial basis weight,

$$g(r_{ij}) \equiv \left[\exp\left(\frac{r_{ij}^2}{2s^2}\right)\right]^{-1}, \tag{2.5}$$

where r_{ij} is the distance from unit i to unit j and s^2 is the variance of the upper triangular portion of the distance matrix. The smoothed vicinity-based propensity score for unit i, $\tilde{e}_i^*$, is the weighted average of the localized propensity scores for all units in its vicinity:

$$\tilde{e}_i^* \equiv \frac{1}{\sum\limits_{j \in \mathcal{V}_i} g(r_{ij})} \sum_{j \in \mathcal{V}_i} g(r_{ij})\, e_j^*, \tag{2.6}$$

where $\mathcal{V}_i$ is unit i's vicinity (here we remind readers that a unit is included in its own vicinity). Smoothed vicinity weights can then be calculated from the smoothed vicinity-based propensity scores as in Equation 2.4.

2.3.4.3 Weights from Coarsened Exact Matching

Another type of ATM weight could be created by partitioning the (possibly transformed) covariate space in some way, creating strata, clusters, or (hyper-)bins. Within each stratum, stratum-based propensity scores and weights can be calculated using logic similar to that used in Section 2.3.4.2. In order for strata produced by a matching or clustering algorithm to be usable in calculating ATM weights, two conditions must hold: there must be no constraints on the number of treated or control units in each stratum, and the algorithm must be able to incorporate some type of caliper.

One algorithm that meets these criteria is Coarsened Exact Matching (CEM) (Iacus et al., 2012), which produces strata using a technique similar to the creation of a multivariate histogram. Iacus et al. provide instructions for using the CEM strata to calculate weights for each subject; their weights can best be described as "local ATT weights," giving each treated unit a weight of one in strata where at least one treated unit and one control unit are present. To create ATM weights instead, we would first calculate a stratum-based propensity score for each stratum $\mathcal{S}$ as the proportion of the units in stratum $\mathcal{S}$ that are in the treated group, then assign this stratum-based propensity score to all units in stratum $\mathcal{S}$. We would then use the stratum-based propensity scores to create weights as in Equations 2.3 and 2.4. In general we would expect CEM-based ATM weights to be more discretized than vicinity weights, because

all units in the same stratum will have the same stratum-based propensity score, whereas under vicinity weighting, each unit could theoretically have a unique vicinity-based propensity score. To create smoother CEM-based ATM weights, we could smooth the stratum-based propensity scores by first repeating the CEM process using several different binning methods and/or coarsenings (a general strategy suggested in Iacus et al. (2012)) and then averaging the stratum-based propensity scores for each unit across the different repetitions. The smoothed stratum-based propensity scores can then be used in the calculation of smoothed CEM-based ATM weights, just as the smoothed vicinity weights are created from the smoothed vicinity-based propensity scores.

2.4 Case study

In this section we illustrate ATM estimation on data originally analyzed by Connors et al. (1996) and publicly available at http://biostat.mc.vanderbilt.edu/DataSets. In contrast to the careful clinical analysis presented by Connors et al., here we use the data for purely illustrative purposes. We consider 2,569 hospital patients from Phase I of the Study to Understand Prognoses and Preferences for Outcomes and Risks of Treatments (SUPPORT), 981 of whom received right heart catheterization (RHC) within the first 24 hours after study entry and 1,588 of whom did not. We are interested in the effect of RHC on average length of stay in the hospital, and we are interested in this effect specifically on the evenly matchable subjects: those subjects, who, in the observed data, occupied a part of the covariate space in which members of the opposite group were at least as plentiful as members of their own group— in other words, subjects who, by all appearances, could have just as easily received the other treatment. For simplicity, rather than using the full set of covariates used by Connors et al., we consider only the following covariates: type of medical insurance, primary disease category (collapsed into eight categories), secondary disease category (collapsed into four categories), presence of neurological diagnosis at admission, do-not-resuscitate status, age, sex, education, SUPPORT model estimate of the probability of surviving two months, Glasgow Coma Score, mean blood pressure, hematocrit, sodium, creatinine, history of renal disease, and history of upper GI bleeding. For purposes of illustration we will assume that these covariates are the only possible confounders. Data-preparation code for the case study can be found in Appendix A (Section 2.6).

Figure 2.3 shows, in red, the absolute standardized mean differences (SMDs) for

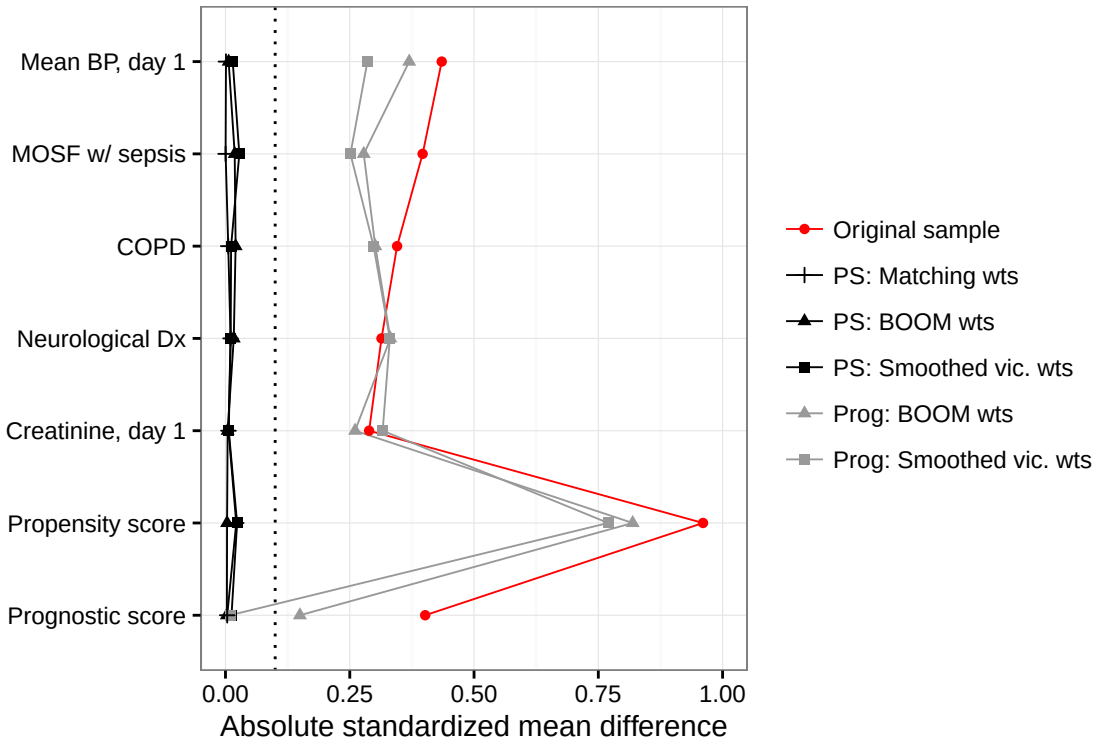

Figure 2.3: Absolute standardized mean differences from case study, for the five baseline covariates with the worst balance in the original sample and for the estimated propensity and prognostic scores, shown for the original sample and for cohorts created using five ATM weighting schemes. Points within a given weighting scheme are connected by a line to facilitate visual tracking. The dotted line at 0.1 marks a degree of imbalance that some researchers consider unacceptable. BP: blood pressure; MOSF: Primary disease of multiple organ system failure; COPD: Primary disease of chronic obstructive pulmonary disease; Dx: diagnosis; PS: propensity score; wts: weights; vic.: vicinity; Prog: prognostic score.

the five baseline covariates with the worst balance in the original sample and also for the estimated propensity and prognostic scores (models given in Appendix B, Section 2.7). The degree of imbalance on all these measures is quite high, suggesting that failure to adjust for important baseline covariates will likely lead to biased estimates of the effect of RHC. Figure 2.4 shows, in red, results from such an unadjusted estimate: a t-test estimates the increase in hospital length-of-stay for patients receiving RHC at about five days, with a 95% confidence interval whose lower limit is well above zero. We will now illustrate several ways of estimating both ATM_{PS} and ATM_{Prog} in this sample and compare the resulting estimates to this unadjusted estimate of treatment effect.

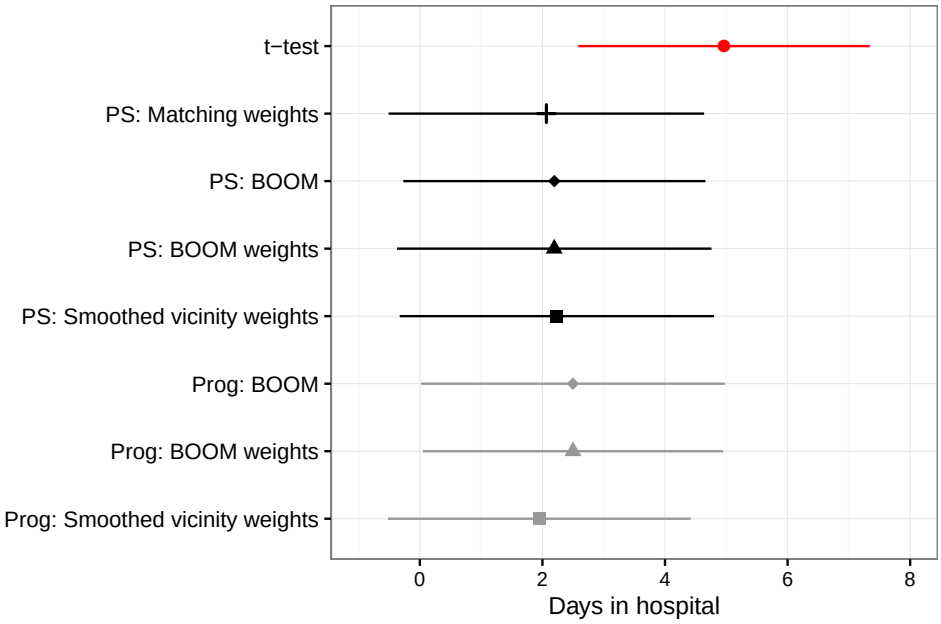

Figure 2.4: Point estimates and 95% confidence intervals from estimation methods used in case study. Point shapes provide no extra information for this plot; they are included to facilitate comparisons between this figure and Figure 2.3. PS: propensity score; Prog: prognostic score.

2.4.1 Case study: Estimating ATM_{PS}

The top panel of Figure 2.5 shows a histogram of the estimated propensity scores in the original sample, giving a more detailed view of the imbalance shown in Figure 2.3. We see that, for propensity scores below about 0.5, the treated subjects are evenly matchable, and in most of the range above this value, the control subjects are evenly matchable. In addition, we see that at the upper end of the propensity score distribution, there are no control subjects at all. Although the recommended approach to this complete lack of overlap would be to prune the dataset before estimation in an attempt to improve the overlap, here we leave the area of nonoverlap as an element of the illustration.

Figure 2.6 shows three of the propensity-score-based weighting schemes discussed in Section 2.3. The top panel shows the closed-form matching weights of Li and Greene (2013) (Section 2.3.3); the middle panel shows the weights generated by the BOOM process (Section 2.3.4.1); and the bottom panel shows the smoothed vicinity weights (Section 2.3.4.2). To generate the BOOM weights, we ran the BOOM process with 10,000 bootstrap resamples, re-estimation of the propensity score within each resample, greedy matching, and a caliper of 0.2 times the standard deviation of the estimated propensity scores in each bootstrap sample. In creating the smoothed

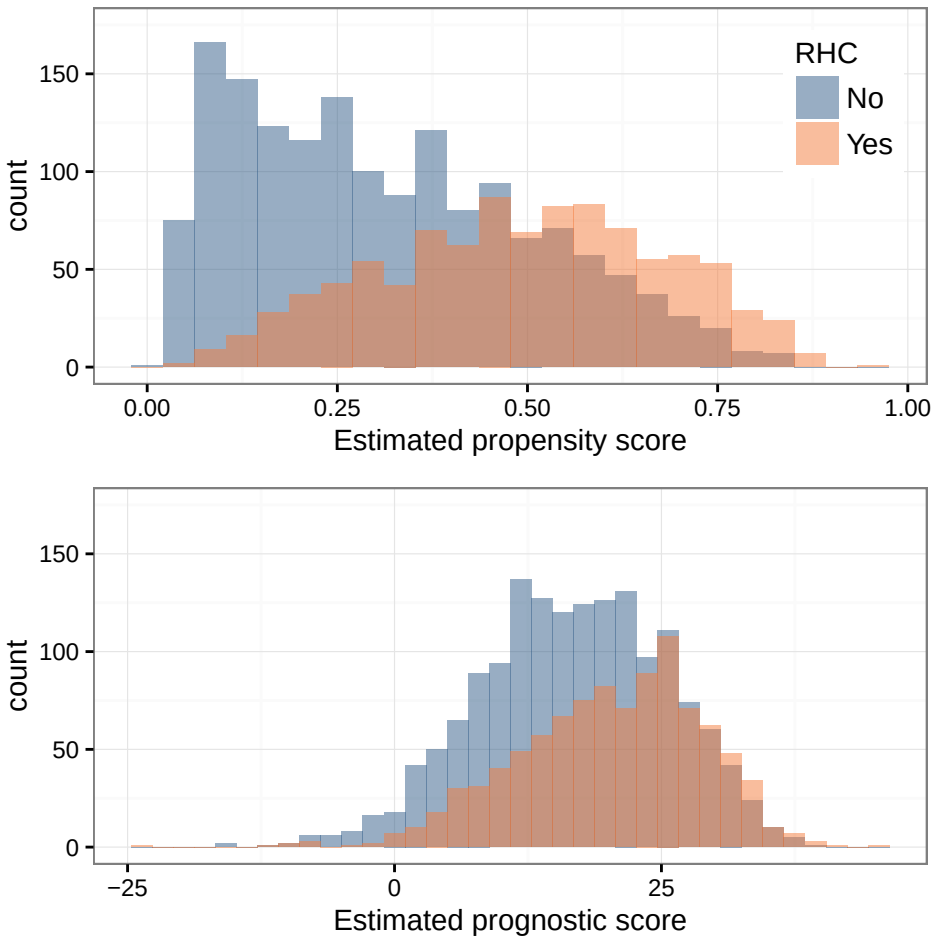

Figure 2.5: Estimated propensity scores and prognostic scores from case study. In each plot, the bin width is equal to the caliper used in the creation of vicinity weights in Sections 2.4.1 (upper plot) and 2.4.2 (bottom plot). RHC= right heart catheterization (the treatment of interest).

vicinity weights, we used a caliper of 0.2 times the standard deviation of the estimated propensity scores in the original sample, or approximately 0.04; for reference, this caliper is used as the bin width in the top panel of Figure 2.5. In comparing the three weighting schemes, we see that they look remarkably similar, with three differences standing out. First of all, the BOOM weights look "fuzzier" than the other two sets of weights. This is because for individuals with the same treatment group and propensity score, there is only one possible matching weight or vicinity weight, but a (generally narrow) range of BOOM weights is possible. Related to this, we see that the BOOM weights have a much smoother transition around propensity score 0.5 than the other weights do. Finally, we see the difference in handling of the treated subject with the highest propensity score (0.94), who is also the sole occupant of the isolated bin in the top panel of Figure 2.5. The matching weight scheme assigns this subject a weight of 0.06; the BOOM weighting scheme assigns this subject a weight of 0.01; and the smoothed vicinity weighting scheme assigns this subject a weight of exactly zero. It is unlikely that this difference in weighting for one subject would make a difference in this case study; we draw the reader's attention to the differences as an illustration of the way the different methods handle subjects in isolated areas of complete nonoverlap.

Given the similarity of the weights assigned under the three methods, we would expect them to create weighted cohorts that are very similar to each other, and indeed they do. The absolute standardized mean differences for the PS-based methods, displayed in black in Figure 2.3, show not only that the degree of marginal balance on the selected covariates and scores is very similar in the three weighted cohorts, but also that all three methods have performed extremely well in terms of balancing the marginal distributions of these variables. One surprising finding is that the BOOM weights balance the propensity score from the original sample just as well as the matching weights and vicinity weights do; because the propensity score was re-estimated in each bootstrap resample, we would not necessarily expect the BOOM weights to balance the original propensity score.

From cohorts with such similar weights and covariate balance, we would expect similar point estimates and confidence intervals for the effect of RHC on hospital length of stay. In addition, because we are not conducting additional covariate adjustment after the weighting, we would expect the point estimate from the BOOM weights to be very close to the point estimate from the BOOM process itself, and we would also expect the confidence intervals from these two methods to be similar as well (Section 2.3.4.1). The point estimates and confidence intervals from the

Figure 2.6: Three propensity-score-based ATM weighting schemes from case study. RHC: right heart catheterization (treatment of interest).

three weighting methods and the BOOM process, shown in black in Figure 2.4, are indeed very similar to each other. All four methods give point estimates of just above two days, with 95% confidence intervals that clearly include zero. Here, rather than using the approach to standard-error estimation suggested by Li and Greene (2013) for the matching weights, we are using the Taylor series linearization approach from the R package **survey** (Lumley, 2014, 2004) for standard error estimates for all three weighted estimators. The point estimates from the four ATM_{PS} methods are consistent with the direction of the estimate from the unadjusted t-test on the full sample, but they are considerably smaller in magnitude.

2.4.2 Case study: Estimating ATM_{Prog}

In some settings, a researcher might be more interested in estimating the effect of treatment among people who are evenly matchable in terms of prognostic score, rather than among people who are evenly matchable in terms of propensity score. The bottom panel of Figure 2.5 shows a histogram of the estimated prognostic scores in the original sample, giving a more detailed view of the imbalance shown in Figure 2.3. We see that the treated subjects are evenly matchable throughout almost the entire range of prognostic scores, but that there are a few areas in which the control subjects appear to be evenly matchable, and a few bins that contain only one treatment type. As in Section 2.4.1, to better illustrate differences between estimation methods, we do not attempt to remove the areas of complete nonoverlap before proceeding with estimation.

Figure 2.7 shows two of the prognostic-score-based weighting schemes discussed in Section 2.3. The upper panel shows the weights generated by the BOOM process (Section 2.3.4.1), and the lower panel shows the smoothed vicinity weights (Section 2.3.4.2). To generate the BOOM weights, we ran the BOOM process with 10,000 bootstrap resamples, re-estimation of the prognostic score within each resample, greedy matching, and a caliper of 0.22 times the standard deviation of the estimated prognostic scores in each bootstrap sample. In creating the smoothed vicinity weights, we used a caliper of 0.22 times the standard deviation of the estimated prognostic scores in the original sample, or approximately two days; for reference, this caliper is used as the bin width in the bottom panel of Figure 2.5. In comparing the two weighting schemes, we see that while the general patterns are similar, they are considerably less closely aligned than the three propensity-score-based schemes in Figure 2.6. As with the propensity-score-based weights, we see that two individuals

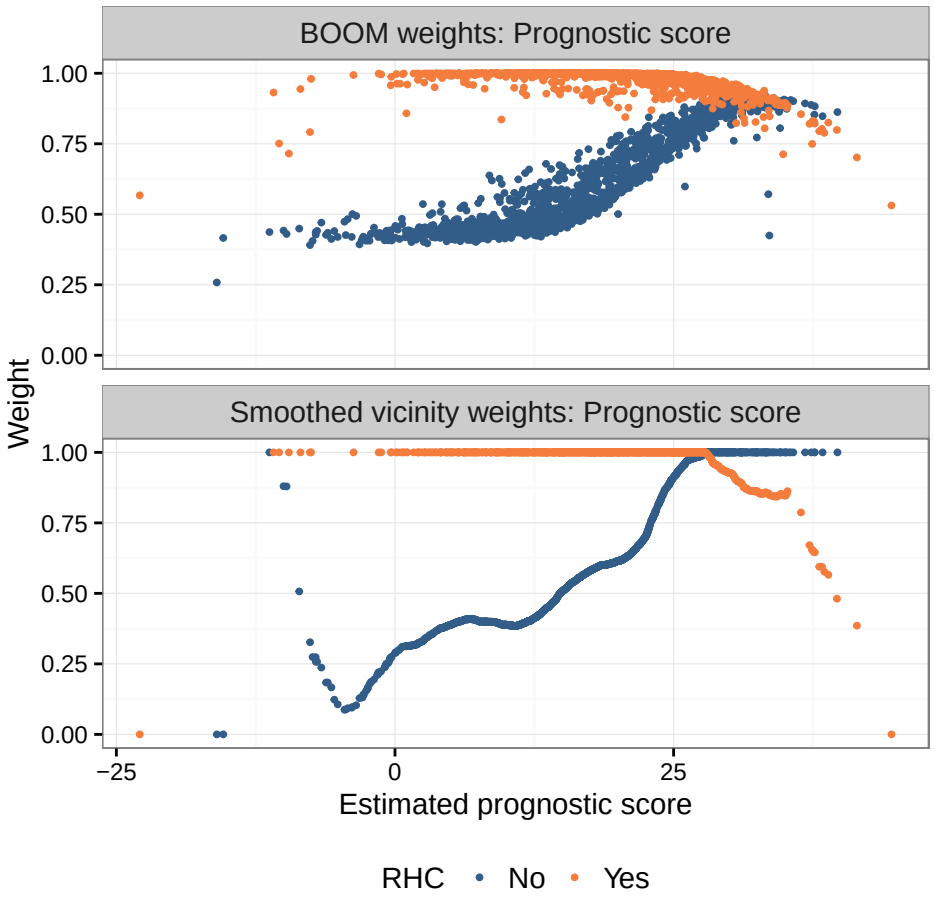

Figure 2.7: Two prognostic-score-based ATM weighting schemes from case study. RHC: right heart catheterization (treatment of interest).

from the same group who have the same prognostic score could have different BOOM weights, but that they will always have identical vicinity weights. Furthermore, we can clearly see differences in the way the two methods handle subjects in isolated regions of nonoverlap in the tails of the prognostic score distribution.

Given the moderate similarity of the weights assigned under the two methods, we would expect the weighted cohorts created by them to be similar, but not as close to each other as the three cohorts created by the propensity-score-based weighting systems. The absolute standardized mean differences in these cohorts, shown in grey in Figure 2.3, bear this out. In addition, Figure 2.3 highlights an important characteristic of prognostic scores: unlike balancing on the propensity score, balancing on the prognostic score does not necessarily balance the marginal distributions of the covariates, even asymptotically. That is, the lack of balance in the baseline covariates

does not represent a failure on the part of the ATM_{Prog} weights— the weights were intended to balance only the prognostic score. As expected, then, the smoothed vicinity weights have produced a cohort with excellent balance on the prognostic score. In contrast, however, while the BOOM weights have greatly improved the balance on the prognostic score, the absolute standardized difference in the weighted cohort is still slightly above 0.10, a level that some researchers would consider unacceptable. As with the implementation of BOOM used in Section 2.4.1, the implementation of BOOM used here re-estimated the prognostic score in each bootstrap resample. In contrast to Section 2.4.1, though, where the BOOM weights nevertheless produced excellent balance on the original-sample propensity score, here the balance is barely adequate. It is possible that, because the prognostic score model is fit using only the control subjects, the coefficient estimates vary more greatly from one resample to another. If a researcher encountered a situation like this and if the volatility of the estimated prognostic scores were a concern, one option would be to run the BOOM process without re-estimating the prognostic scores in each bootstrap resample.

From weighted cohorts that appear to be similar but not identical in terms of marginal covariate distributions, we would expect similar but not identical point estimates and confidence intervals for the effect of RHC on hospital length of stay. In addition, because we are not conducting additional covariate adjustment after the weighting, we would expect the point estimate from the BOOM weights to be very close to the point estimate from the BOOM process itself, and we would also expect the confidence intervals from these two methods to be similar as well (Section 2.3.4.1). The point estimates and confidence intervals, show in grey in Figure 2.4, bear these expectations out: the two weighting methods yield similar, but clearly not identical, results, and the results from the two BOOM methods are extremely close to each other. Comparing these results to those from the PS-based methods, we see that the BOOM-based prognostic-score methods yield point estimates slightly higher than those from the propensity score methods, and the smoothed vicinity ATM_{Prog} weights yield a point estimate that is slightly lower. While the confidence intervals for the BOOM ATM_{Prog} estimate and the BOOM-weighted ATM_{Prog} estimate technically do not include zero, for the purposes of this illustration we consider the estimates and confidence intervals from all the propensity-score and prognostic-score methods to be very close. All of the ATM estimation methods have produced results that are more tempered than the estimate from the unadjusted t-test.

2.5 Discussion

2.5.1 The value of the ATM

In describing what we would now refer to as ATM estimators, Li and Greene (2013, p. 227) write that they "can be interpreted as emphasizing patients for whom there is greater equipoise; that is, patients whose characteristics are such that physicians are roughly equally likely as not to assign treatment. ... [T]hese patients may often be of most interest when contrasting treatment alternatives." Li and Greene are not alone in their belief that these patients have special value; in earlier work, Rosenbaum (2012, pp. 57–58) advocates the study of these patients, whom he calls "marginal patients":

> Study of the marginal patient may guide treatment of the marginal patient, and it may also serve to initiate questions about a consensus of opinion concerning the treatment for patients not deemed marginal. That is, study of the marginal patient may shift where that margin is, so that over a period of time, a sequence of studies may gradually shift a consensus. Study of the marginal patient is attractive in the specific but not inconsequential sense that there is the realistic hope of locating patients who look similar in terms of observed covariates yet who received different, competing treatments.

In focusing our inference on the evenly matchable units, we are choosing the units for whom the least amount of model extrapolation is required, for both the propensity score model and the outcome model. While other estimands may also be of interest, researchers might not necessarily have the data to estimate them in a given study. In contrast, the ATM can often be well estimated with the available data.

2.5.2 Choosing among ATM estimation strategies

In this paper we have discussed several ATM techniques that can be used regardless of the distance measure or covariate transformation of interest: caliper pair matching, bagged one-to-one matching (BOOM), BOOM weights, vicinity weights, and CEM weights. Each of these techniques has advantages and disadvantages; there is no single best technique for estimating the ATM.

Of the techniques, caliper pair matching stands alone in that it creates an easily identified, unweighted cohort that is easy for audiences with limited statistical training to comprehend; unfortunately, it also has the potential to discard large numbers of

subjects and thus to be much less statistically efficient than the other approaches. In studies where bootstrapped matching is computationally feasible, researchers who want to use a more statistically efficient method than caliper pair matching can choose between BOOM estimation and estimation via BOOM weights. Several factors may make BOOM weights the more appealing of these two options. At this time, researchers might find that an analysis using a long-accepted method— weighting— is better received than an analysis that relies solely on a new technique. Furthermore, the number of bootstrap resamples needed for accurate standard-error estimation in the BOOM process appears to be much higher than the number of resamples needed to obtain a reasonable set of BOOM weights. Thus a researcher may prefer to run the BOOM algorithm with a lower B (number of resamples) and then use the resulting weights in a weighted analysis, rather than running the BOOM algorithm with a high B to get a bagged estimate with a stable standard error estimate. On the other hand, when imputation of missing data is incorporated into the BOOM process, it is possible that standard error estimates for the BOOM estimator will be more reliable than standard error estimates from analyses using BOOM weights. Further research is needed in this area.

In samples in which bootstrapped matching is not computationally feasible, smoothed vicinity weights and CEM-based ATM weights can be used. While these approaches can technically be used in data sets of any size, smaller datasets are likely to yield highly discretized vicinity weights and CEM-ATM weights even after smoothing, and so at present, we recommend that these methods be reserved for large datasets, although further research is needed in this area. In choosing between vicinity weights and CEM-ATM weights, researchers should consider whether a binning approach is appropriate for the particular (possibly transformed) dataset being studied.

When the ATM_{PS} is of interest, matching weights are also an excellent option. Li and Greene argue for the use of matching weights over PS caliper pair matching; we now consider them in relation to the new estimation methods presented here. For studies in which bootstrapped matching is computationally feasible, the pros and cons of matching weights versus BOOM weights are discussed above in Section 2.3.4.1; similar considerations apply to the decision between matching weights and the standard BOOM estimator. In general, BOOM and BOOM-weighted estimation are more flexible processes that allow repeated fitting of the propensity score model and automatic exclusion of data points outside the area of common support, but a key advantage of matching weights over either BOOM weights or BOOM estimation is that estimation via matching weights is a much quicker process. If bootstrapped matching is not

computationally feasible, the choice between matching weights and propensity-score-based vicinity weights is one of philosophy or preference for the handling of units in isolated areas of nonoverlap, although proper pruning of the dataset before the estimation of propensity scores would probably eliminate this distinction. Due to the necessarily arbitrary binning inherent in Coarsened Exact Matching on the propensity score, we see no advantage in the use of CEM-ATM weights over matching weights when the ATM_{PS} is of interest.

2.6 Appendix A. Data-preparation code for case study

```
# mydat1 is the raw RHC data from the web
mydat1 <- within(mydat1, {
    swang1.01 <- ifelse(swang1 == "RHC", 1, 0)
    days.in.hosp <- dschdte - sadmdte
    days.alive <- dthdte - sadmdte
    los <- pmin(days.in.hosp, days.alive, na.rm= TRUE)

    # this variable is messing up latex
    ninsclas <- gsub("&", "+", ninsclas, fixed= TRUE)

    # using the SAS default origin seems to work
    admitDate <- as.Date(sadmdte, origin= "1960-01-01")
    phase1 <- admitDate <= as.Date("1991-12-31")

    cat2 <- ifelse(is.na(cat2), "None Listed", cat2)
})

mydat1.1 <- mydat1[mydat1$phase1 == TRUE, ]

mydat <- mydat1.1

cat1dat <- data.frame(model.matrix(~ cat1 - 1, data = mydat))
cat1vars <- names(cat1dat)
mydat <- cbind(mydat, cat1dat)

cat2dat <- data.frame( model.matrix(~ cat2 - 1, data = mydat))
cat2vars <- names(cat2dat)
mydat <- cbind(mydat, cat2dat)

ninsclasdat <- data.frame(model.matrix(~ ninsclas - 1, data = mydat))
ninsclasvars <- names(ninsclasdat)
mydat <- cbind(mydat, ninsclasdat)

neurodat <- data.frame(model.matrix(~ neuro - 1, data = mydat))
neurovars <- names(neurodat)
mydat <- cbind(mydat, neurodat)

dnr1dat <- data.frame(model.matrix(~ dnr1 - 1, data = mydat))
dnr1vars <- names(dnr1dat)
mydat <- cbind(mydat, dnr1dat)

sexdat <- data.frame(model.matrix(~ sex - 1, data = mydat))
```

```
sexvars <- names(sexdat)
mydat <- cbind(mydat, sexdat)
```

2.7 Appendix B. Formulas for case study

```
# Propensity score formula
ps.form <- swang1 ~
    #ninsclas: using all categories
        ninsclasMedicare +
        ninsclasMedicare...Medicaid +
        ninsclasNo.insurance +
        ninsclasPrivate +
        ninsclasPrivate...Medicare +
    #cat1: collapsing some categories
        cat1ARF +
        cat1CHF +
        cat1Cirrhosis +
        #cat1Colon.Cancer +
        cat1Coma +
        cat1COPD +
        #cat1Lung.Cancer +
        cat1MOSF.w.Malignancy +
        cat1MOSF.w.Sepsis +
    #cat2: collapsing some categories +
        #cat2Cirrhosis +
        #cat2Colon.Cancer +
        #cat2Coma +
        #cat2Lung.Cancer +
        cat2MOSF.w.Malignancy +
        cat2MOSF.w.Sepsis +
        cat2None.Listed     +
    #neuro +
        neuroYes +
    #dnr1 +
        dnr1Yes +
    rcs(age, 4) +
    sexFemale +
    edu +
    rcs(surv2md1, 4) +
    scoma1 + # not really continuous. do not use spline
    rcs(meanbp1, 4) +
    rcs(hema1, 4) +
    rcs(sod1, 4) +
```

```
    rcs(crea1, 4) +
    renalhx +
    gibledhx

# Prognostic score formula
prog.form <- update.formula(ps.form, los ~ .)
```

Chapter 3

Visual Pruner: Transparent and Flexible Cohort Selection for Observational Studies

3.1 Introduction

3.1.1 Why prune?

Observational studies, commonly conducted in fields including medicine, public health, epidemiology, economics, and political science, enable researchers to assess the effect of a treatment or exposure in situations where random assignment to study groups would be unethical or infeasible. An important concern in observational studies is whether the groups being studied are comparable in terms of baseline covariates (variables measured before the treatment or exposure) that might affect the outcome of interest. This comparability, or *balance*, has two components: in a perfectly balanced study, the covariate distributions in the two groups overlap completely (have common support) and have the same shape (Ho et al., 2007). Poor balance in pre-treatment covariates can lead to bias and model dependence in treatment effect estimates (Ho et al., 2007). Popular approaches to mitigating imbalance in observed baseline covariates include matching on exact covariate values or on a multivariate distance measure; matching, weighting, stratification, or regression adjustment using the estimated propensity score, or probability of being in the treated group given the observed covariates (Rosenbaum and Rubin, 1983); and statistical modeling using the covariates.

When applied before any of these techniques, *pruning*, or removing some units from the available data to select a smaller study cohort, can further reduce covariate imbalance, leading to less model-dependent (Ho et al., 2007) and thus more convincing results. Pruning is especially helpful in ensuring common support but can also improve the similarity of the distributions in the newly restricted range. Furthermore, pruning can be essential to the development and evaluation of the propensity score model in analyses that use propensity scores (Ho et al., 2007; King and Zeng, 2006), as discussed further below.

3.1.2 Current pruning methods

Although we present pruning here as a technique to be applied before the use of other methods, some methods for observational studies incorporate pruning implicitly. For example, if the original sample has fewer treated units than control units, one-to-

one matching will discard some units from the original sample as part of the matching process itself. While this "implicit pruning" can certainly be successful in producing a sample with better covariate balance, the resulting sample will have been selected mainly by the matching algorithm, rather than by the researcher's own explicitly-stated inclusion criteria. For studies where the covariate distributions have limited overlap, we join with Ho et al. (2007) and other authors in recommending at least the consideration of some explicit pruning before the use of one-to-one matching and other techniques that algorithmically eliminate study units.

In particular, with the goal of having clearly defined study inclusion criteria, we join with other authors in recommending not only the consideration of explicit pruning before proceeding with other techniques, but also that pruning, if necessary, be based on the distribution of the baseline covariates. In contrast, some analysts prune based on the distribution of the estimated propensity scores, eliminating observations outside the region of overlap of the estimated scores before proceeding with matching or weighting. This approach is less than ideal for two reasons. First of all, as King and Zeng (2006) point out, it is hampered by "a fundamental problem of infinite regress: we cannot use the propensity score to identify regions of extrapolation until we can verify that the estimated propensity score is valid, but we cannot verify the validity of the estimated propensity score until we have first removed the regions requiring extrapolation." More fundamentally, Rosenbaum argues that it may be preferable to select a study cohort based on covariate values rather than on propensity scores: "A population defined in terms of [the propensity score] is likely to have little meaning to other investigators, whereas a population defined in terms of one or two familiar covariates ... will have a clear meaning" (Traskin and Small, 2011; Rosenbaum, 2010, p. 86).

Several covariate-based pruning approaches are currently available. The hyper-rectangle approach (Porro and Iacus, 2009) yields a simple, easily understandable set of inclusion criteria, but because of its simplicity and lack of flexibility, in some settings this approach can remove too many subjects on the edges of the restricted covariate space while leaving large areas of imbalance in the interior. The convex-hull approach (King and Zeng, 2006), while leaving fewer large areas of imbalance in the interior of the covariate space, can also remove too many subjects on the edges and does not result in easily understandable inclusion criteria. In recent work, Traskin and Small (2011) propose an approach to generate transparent covariate-based inclusion criteria derived from the optimal cohort selection methods developed by Rosenbaum (2012) (a matching algorithm that prunes implicitly while optimizing

the sample size and the sum of the distances between subjects) and Crump et al. (2009) (a propensity-score-based pruning technique that optimizes the precision of the treatment-effect estimator). Traskin and Small's approach uses classification trees to translate the cohort selected by one of the optimal methods into a similar cohort that can be described in terms of covariates rather than in terms of algorithms or propensity scores.

3.1.3 **Visual Pruner**'s contribution

Because classification trees produce rules based on interactions between variables, the criteria yielded by Traskin and Small's approach can be quite complex. In some settings, this complexity might be unavoidable. In other settings, though, an analyst might prefer to have simpler inclusion criteria, even if the resulting cohort is farther from optimal in terms of the quantities optimized by Rosenbaum (2012) or Crump et al. (2009). For observational studies in which analysts would like to have simple inclusion criteria while still taking advantage of information in the propensity scores, the web application **Visual Pruner** allows analysts to make pruning decisions guided by both the univariate distributions of the covariates and the distribution of the estimated propensity scores, to create data-driven, transparent inclusion criteria.

In addition to its pruning capabilities, **Visual Pruner** has additional features that can be especially valuable in studies where the analyst will go on to use estimated propensity scores in matching or inverse probability weighting (IPW). Austin and Stuart (2015) recommend that analysts using IPW do extensive checking of covariate balance in the weighted sample, and they endorse the advice given by Rosenbaum and Rubin (1984) that development of the propensity score model may need to be an iterative process. Unfortunately, Austin and Stuart (2015) found that many published studies using IPW do not contain any evidence of balance checking or iterative propensity score development. **Visual Pruner** makes it easy for analysts to follow Austin and Stuart's excellent recommendations for best practice.

Visual Pruner can be readily incorporated into a reproducible-research workflow. After pruning a dataset with **Visual Pruner**, analysts can download files containing the final propensity score model and inclusion criteria and then proceed with the remainder of the statistical analysis as planned. Use of the app can improve both the credibility and the transparency of observational studies.

3.2 Using the app

The app, freely available at http://statcomp2.vanderbilt.edu:37212/VisualPruner, is built using the R **shiny** framework (R Core Team, 2015; Chang et al., 2015). Analysts running the app from the Web do not need R or its packages. Analysts wishing to run the app on their local machine can either copy the app's source files, displayed on the app's R tab and also available at https://github.com/LaurenSamuels/VisualPruner, into a local directory and run them with **shiny**'s `runApp()` function, or use **shiny**'s `runGitHub()` function, with arguments `repo= "VisualPruner", username= "LaurenSamuels"`.

Using the app consists of moving (in most cases, iteratively) through five steps, each conducted on a separate tabbed page within the app. The steps are described in detail in the subsections below and summarized in Table 3.1. Additional information is contained in Notes sections at the bottom of some of the tabbed pages, as well as on the app's About tab.

3.2.1 Tab 1 ("Upload"): Upload data

Visual Pruner accepts datasets in .csv or .rds format. After the analyst uploads a file, the app displays the number of columns and rows as a means of verification. The app then presents a list of all dichotomous variables in the dataset (numeric or character), and the analyst chooses one to use as the treatment indicator. The app displays the levels of the chosen treatment indicator.

Rather than uploading a dataset, new users may prefer to select the "Use example dataset" radio button. An example dataset consisting of 1000 observations of seven variables (a treatment indicator and six pre-treatment covariates) will then be generated. This dataset will be used for illustration in Section 3.4.

3.2.2 Tab 2 ("Specify"): Specify a propensity score model

Before specifying a propensity score model, the analyst specifies how the app should handle missing values in fitting the propensity score model. The default is to do a single imputation, replacing any missing values with the univariate mean or mode for purposes of model fitting. This approach, together with the use of missingness indicators (see below), can result in unbiased treatment effect estimates under certain conditions; see Cham and West (2016) and Rosenbaum (2010, pp. 193–194, 240–242) for more details. Alternatively, the analyst can choose to estimate propensity scores for only those subjects who have no missing values for the variables used in the model.

Table 3.1: Summary of **Visual Pruner**'s seven tabbed pages.

Tab	Key Actions	Notes
Upload (§ 3.2.1)	Upload data Select treatment indicator	Accepted formats: .csv, .rds
Specify (§ 3.2.2)	Choose method for handling missingness in model-fitting Specify right-hand side of propensity-score model	
Prune (§ 3.2.3)	Choose variables to view and restrict Choose cutoff for discreteness Modify graphical parameters (optional) Brush propensity-score histogram (optional)	
	View displays for selected covariates	Plots will not appear until variables have been selected and "(Re-)make graphs ..." button has been clicked.
	Make pruning decisions	Click one or both pruning buttons (above first covariate display) to apply changes.
Compare (§ 3.2.4)	View SMD plot	Plot will not appear until variables have been selected, and "(Re-)make graphs ..." button has been clicked, on the Prune tab.
Download (§ 3.2.5)	Download or copy final inclusion criteria and propensity score model	
About	Read more about the app	
R	See R session information and source code	

Regardless of which option is chosen, the choice applies to only those variables used in the propensity score model, and it applies only during the process of model fitting. No pruning of the dataset is done at this stage.

The analyst then specifies the right-hand side of a propensity score model, which will be fit using binary logistic regression with the `lrm()` function from the **rms** package (Harrell Jr., 2015). The app displays a list of the variables in the dataset for convenience. The model formula can use the full range of base R and **rms** features, including interactions and restricted cubic splines. (Users unfamiliar with R can find a guide to R formulas at https://stat.ethz.ch/R-manual/R-devel/library/stats/html/formula.html. A link to the guide is also available in the app.) If the analyst has chosen to impute values for variables with missing values, the app automatically generates missingness-indicator variables that can be included in the propensity score formula (here "missingness" refers to the state in the original dataset, before imputation). Dynamically generated directions for using the new variables will appear in the app.

After the analyst clicks the "I have finished typing" button, the app checks the basic syntax of the formula and the validity of the variable names and then attempts to fit the model. If the model can be fit, the app displays histograms of the estimated propensity scores on both the probability and the log-odds scale; while the probability scale is often intuitively easier to grasp, the log-odds scale is more often used in matching applications and allows enhanced viewing of the tails of the distribution.

3.2.3 Tab 3 ("Prune"): Refine inclusion criteria

The Prune page consists of an upper section and a lower section, separated by two buttons that perform the actual pruning. The upper section contains general controls and displays, while the lower section contains individual controls and displays for each of the selected covariates. In the first step in the upper section, the analyst must choose variables to view and restrict; in most cases, analysts will want to select at least those variables used in the propensity score model, but this is not required. The analyst's next choice involves the display of discrete numeric variables. By default, when the covariate graphs are produced, numeric variables will be displayed as discrete variables if they have fewer than ten unique values in the original dataset. After selecting variables and possibly modifying the cutoff for discreteness, the analyst clicks the "(Re)-make graphs using updated variable list and/or discreteness preferences" button. Two other graphical preferences can be modified here: the

opacity (alpha) and the plotting symbol size. These (as well as the other two) can be updated at any time; the only difference is that the first two require the click of the "(Re)-make graphs..." button in order to take effect.

In addition to these controls, the upper portion of the Prune page has two key displays: a sample-size table, and a brushable histogram of the estimated propensity scores. Analysts can brush a portion of this plot, and the units that have propensity scores within that range will be highlighted in the covariate plots below. This plot also contains the legend that is used for all plots on the page. The logit scale is used for this plot as well as for the propensity scores in the plots displayed in the lower section of the page.

The bottom part of the Prune page contains a section for each of the variables selected for viewing and restriction. Each section consists of a multi-panel plot, a pruning tool, and a percent-missing table. For continuous variables, the main panel in the multi-panel plot is a scatterplot, with the covariate values on the x-axis, the estimated propensity scores on the y-axis, and a loess curve. Above the main panel is a histogram of the covariate. If, on the Specify page, the analyst chose not to estimate propensity scores for units with missing values for the variables in the propensity score model, the histogram may contain observations that are not included in the scatterplot below; if this is the case, a message will appear. To the right of the main panel is a stripchart showing the propensity score values for units with missing values for this particular covariate, if applicable. The barchart in the upper right panel displays the counts of missing values by group. For discrete variables, the layout of the multipanel plot is similar, except that the main scatterplot is replaced by a stripchart, and the histogram above the main panel is replaced by a barchart. Rather than showing a loess smooth, the stripcharts for discrete variables contain horizontal lines marking the mean propensity score value within each level.

In addition to the multi-panel plot, the section for each variable contains a two-part pruning tool and a percent-missing table. The first part of the pruning tool consists of either text boxes for the maximum and minimum (for continuous variables) or checkboxes for each level (for discrete variables). The second part is a choice of whether to keep (the default) or exclude units with missing values for the variable. The analyst can make pruning decisions for one variable at a time, or for several variables at once, restricting each one to an appropriate range or set of values. In order for the pruning decisions to take effect, the analyst must click one of the two buttons that separate the top part of the Prune page from the bottom part. The "Update covariate graphs to reflect pruning choices" button applies the pruning decisions, then

updates the covariate graphs to reflect the pruned sample; in addition, the sample-size table at the top of the page will also be updated. The "Recalculate PS for pruned sample (will also update all graphs)" button will first apply the pruning criteria, then re-estimate the propensity scores on the pruned sample, then update the graphs and the sample-size table. Clicking this second button is the approach we recommend, but the first one may be useful as part of an incremental approach with larger datasets. Pruning decisions can be modified multiple times, and the analyst can also go back to the Specify tab and change the form of the propensity score model multiple times based on insights gained from the plots and tables.

3.2.4 Tab 4 ("Compare"): View the standardized mean difference plot

While the Prune tab provides a detailed graphical display for each selected covariate, the Compare tab shows a condensed summary of one aspect of the covariate balance between the two groups, in the form of an absolute standardized mean difference (SMD) plot for the selected variables. The absolute standardized mean differences, calculated here using the **tableone** package (Yoshida and Bohn, 2015), are the absolute differences in the group means, scaled by a sample-size-neutral pooled standard deviation (Flury and Riedwyl, 1986; Austin, 2009*b*; Yang and Dalton, 2012). Analysts may wish to consult the Compare tab both before and after making pruning decisions. Before any pruning has been done, the SMD plot contains a single set of points (optionally connected by a line for clarity), that for the original sample. (Note that nothing will be displayed until the analyst chooses variables at the top of the Prune page.) The dotted vertical line at 0.1 marks a degree of imbalance that some researchers consider to be unacceptable (Austin, 2009*a*). After the first pruning decision (once the "Update..." or "Recalculate..." button is clicked on the Prune page), the SMD plot on the Compare page will be updated to include an additional set of points, this one for the pruned dataset. This line will be updated each time the inclusion criteria are updated.

Analysts who are considering conducting a weighted analysis after pruning can also choose to add points for three types of weighted samples to the SMD plot: ATE weighting, used to estimate the average treatment effect on a sample or population that resembles the pruned sample; ATT weighting, used to estimate the average treatment effect on a sample or population that resembles the treated units in the pruned sample, and ATM weighting, used to estimate the average treatment effect on a sample or population that resembles the evenly matchable units from both

groups in the pruned sample. ATM weighting uses weights introduced as "matching weights" by Li and Greene (2013); the weights are discussed further in a forthcoming manuscript (Samuels and Greevy, Jr., 2016*b*). All weighting is conducted using the **survey** package (Lumley, 2014, 2004).

In general it is important to consider standardized mean differences for higher-order terms and interactions (Austin and Stuart, 2015), as well as for missingness indicators. We hope to add automatic generation and inclusion of these variables in the future, but in the meantime we recommend that analysts add them to their datasets before importing so that they can be viewed within the app.

3.2.5 Tab 5 ("Download"): Download the inclusion criteria and propensity score model

The analyst can iterate through the steps of specifying a propensity score model, pruning the dataset, and viewing the standardized mean difference plot many times and in any order after the initial model specification and selection of variables to view. Ideally, at some point the dataset will have a degree of covariate balance that the analyst finds acceptable.

Rather than providing a final version of the dataset, **Visual Pruner**'s Download page provides downloadable (or copiable) R code for the study's inclusion criteria and the final propensity score model, in order to facilitate reproducible research. The R code used here is simple enough that users of any statistical software should be able to understand it and translate it if necessary.

3.3 After using the app

In some cases, pruning according to the criteria downloaded from **Visual Pruner** will be sufficient to create treatment groups that have good balance on all observed covariates, in terms of both the overlap and the shape of the distributions. In most cases, however, after the analyst applies the inclusion criteria to the data, the groups will not be perfectly balanced. The pruning can then be followed by matching or weighting on the propensity score (estimated on the pruned sample, using the final formula downloaded from the app) or by other methods, alone or in combination with statistical modeling (Ho et al., 2007). Note that for analyses using propensity score estimation, if the estimation process used within the app relied on the imputation of missing values, the values will need to be re-imputed before or as part of the propensity-score estimation process. Similarly, if the final propensity score model con-

tains missing-value indicators, these variables will need to be added into the pruned dataset before propensity-score estimation if they were not already present in the data.

3.4 Illustrations

In this section we show several plots from the app, using the example dataset included with the app for illustration. The treatment indicator variable in this dataset is `exposed`, with possible values "Yes" (300 subjects) and "No" (700 subjects). Six baseline covariates are also included: `age` (age in years), `height_ft` (height in feet), `systolicBP` (systolic blood pressure in mmHg), `gender`, (gender: female/male/other), `smoker` (smoking status: current/former/never), and `ABO` (ABO blood type: A/B/AB/O). Some of the variables have missing values.

3.4.1 Illustrations from the Prune tab

Our first illustrations come from the Prune tab. Prior to visiting the Prune tab, on the Specify tab we left unchanged the default method of handling missing values, and we specified the following right-hand side for our initial propensity score model: `age + height_ft + systolicBP + gender + smoker + ABO`. The first plot shown on the Prune tab is the plot of estimated propensity scores, Figure 3.1. Although this plot is brushable in the app (and linked to the other plots on the Prune tab), we do not illustrate this feature here.

A section for each selected variable, consisting of a graphical display, a pruning tool, and a percent-missing table, appears below the propensity score plot on the Prune tab. Figure 3.2 shows a screenshot of the full section for age. In Figures 3.3–3.5, we zoom in on the graphical displays from the sections for the continuous variables in the dataset, before any pruning or any modification of the propensity score model, and give possible interpretations and suggested actions. Figure 3.6 does the same for one of the categorical variables. Section 3.2.3 has more information about these displays in general.

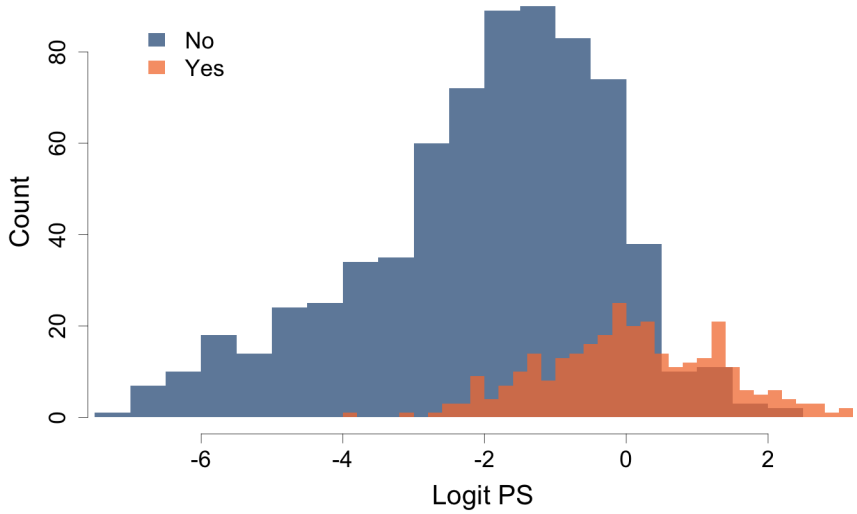

Figure 3.1: Estimated logit propensity scores from the original model, as shown in the Prune tab. The legend used for this plot, showing "No" and "Yes" for the levels of the treatment indicator variable, `exposed`, is the legend used for the plots that follow as well.

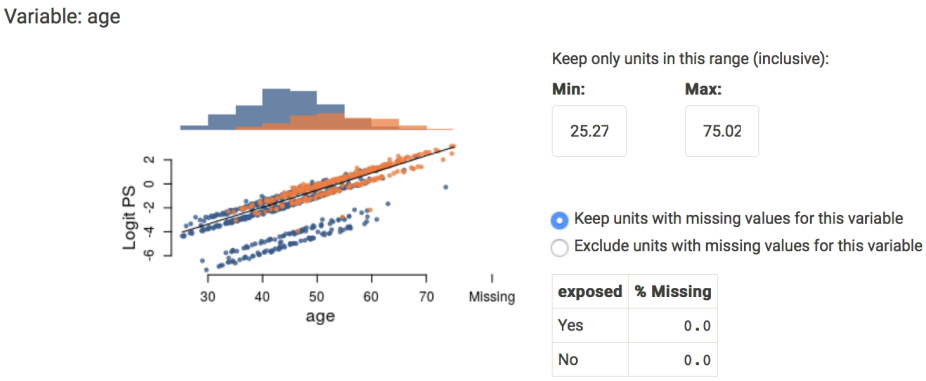

Figure 3.2: Screenshot of the full section for `age` (age in years) from the Prune tab, before any pruning or any modification of the propensity score model. The left half of the section contains the graphical display, and the right half contains the two-part pruning tool (top) and the percent-missing table (bottom). In Figure 3.3 we zoom in on the graphical display from this screenshot and give possible interpretations and suggested actions.

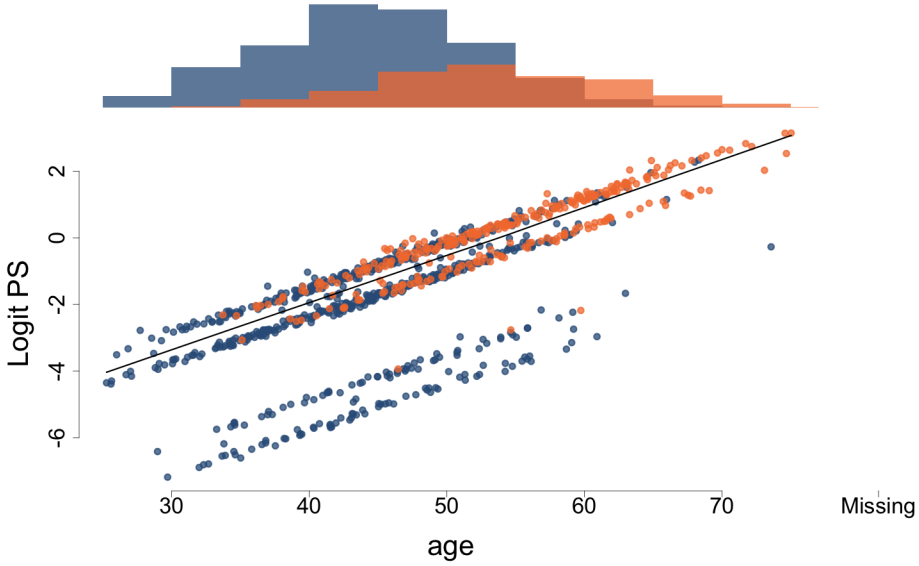

Figure 3.3: First graphical display for `age` (age in years), from the Prune tab. The screenshot in Figure 3.2 gives the visual context for this display. Both the scatterplot and the histogram show a lack of overlap, as well as a difference in the shapes of distributions within the region of overlap: the subjects in the exposed group (orange) are in general older than the subjects in the unexposed group (blue). The scatterplot and the loess curve also show the strong relationship between age and the estimated propensity scores. Analysts unwilling to extrapolate into regions containing subjects from only one group can use the app's pruning tool to restrict the study's inclusion criteria to a narrower range of ages, for example 35–65. Note, however, that in contrast to `height_ft` (Figure 3.4), where simple pruning can completely balance the groups, `age` will still be imbalanced even after use of the app's pruning tool, due to the difference in the shapes of the distributions in the two groups. The empty panels above the label "Missing" show that there are no subjects with missing values for `age`.

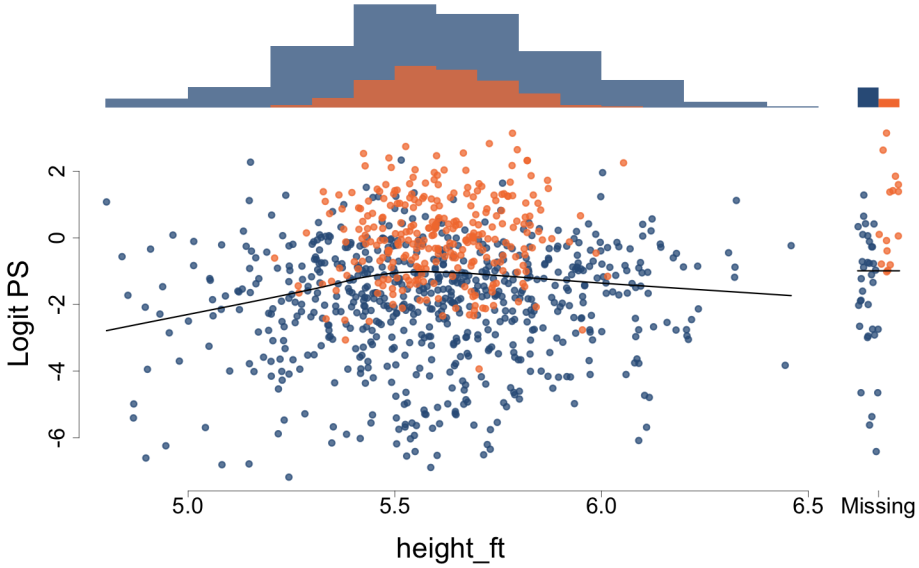

Figure 3.4: First graphical display for `height_ft` (height in feet), from the Prune tab. Both the scatterplot and the histogram show a lack of overlap, with the exposed group (orange) having a much narrower range than the unexposed group (blue), but we see similarly shaped distributions within the region of overlap. Note that the initial propensity score model, in which height is included using a single linear term, does a very poor job of estimating the probability of treatment for subjects in the outer portions of the range of heights. Analysts unwilling to extrapolate into regions with no exposed subjects can use the app's pruning tool to restrict the study's inclusion criteria to a narrower range of heights, for example 5.25 ft–6 ft. The loess curve suggests that within the region of overlap, height has only a weak association with the estimated propensity scores. The panels on the right, together with the percent-missing table for this variable (not shown), suggest that for this variable, missingness does not differ by treatment status.

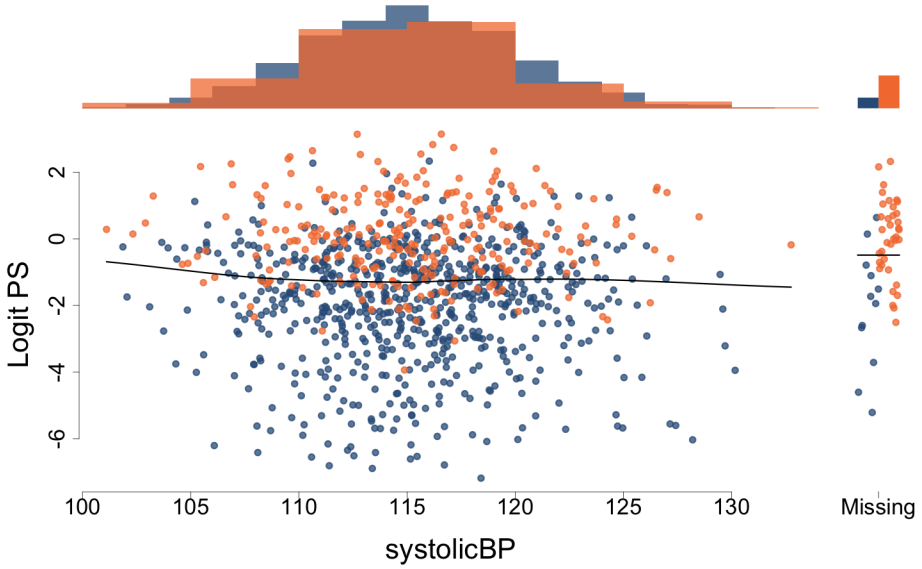

Figure 3.5: First graphical display for `systolicBP` (systolic blood pressure, in mmHg), from the Prune tab. For this variable, both the scatterplot and the histogram show good overlap and similarly shaped distributions. The loess curve suggests that there is little to no relationship between this variable and the estimated propensity scores. Looking at the missing-value plots on the right, together with the percent-missing table (not shown), we see that the proportion of subjects missing this variable in the exposed group (orange) is much higher than that in the unexposed group (blue). As a next step, we recommend returning to the Specify tab and adding a missingness-indicator variable for `systolicBP` into the model.

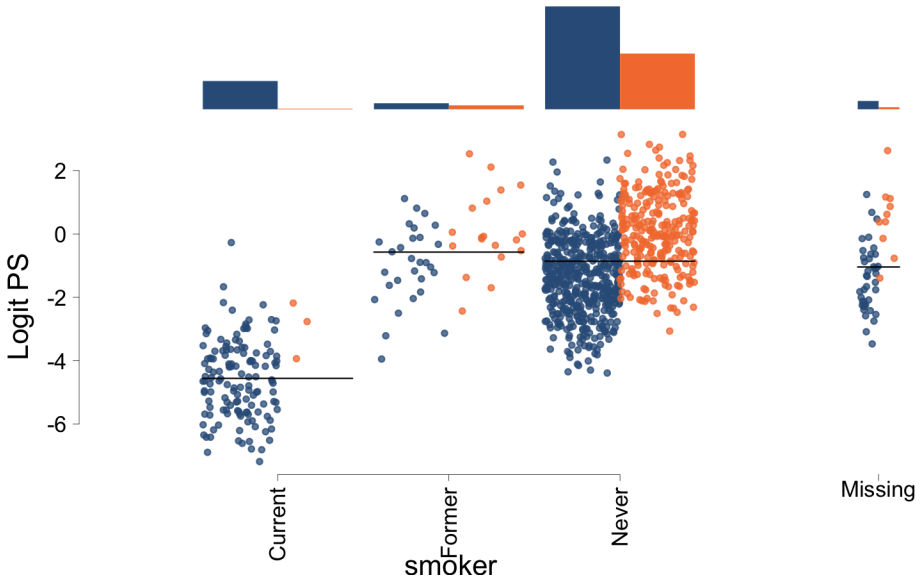

Figure 3.6: First graphical display for `smoker` (smoking status), from the Prune tab. We see in both the main stripchart and the main barchart that smoking status differs heavily by exposure group. Most noticeably, we see that there are only three current smokers in the exposed group (orange). In deciding whether to exclude current smokers from the study, an analyst would need to weigh the loss of power from eliminating these subjects against the risks inherent in making inferences about the effect of exposure in current smokers when there are so few current smokers in the exposed group.

3.4.2 Illustrations from the Compare tab

As discussed in Section 3.2.4, analysts may wish to alternate consulting the detailed covariate displays on the Prune tab and the condensed summary display on the Compare tab. Figure 3.7 shows the first absolute standardized mean difference (SMD) plot for the original sample from the Compare tab. Figure 3.8 shows one possible SMD plot after updating the propensity score model and trying some initial pruning.

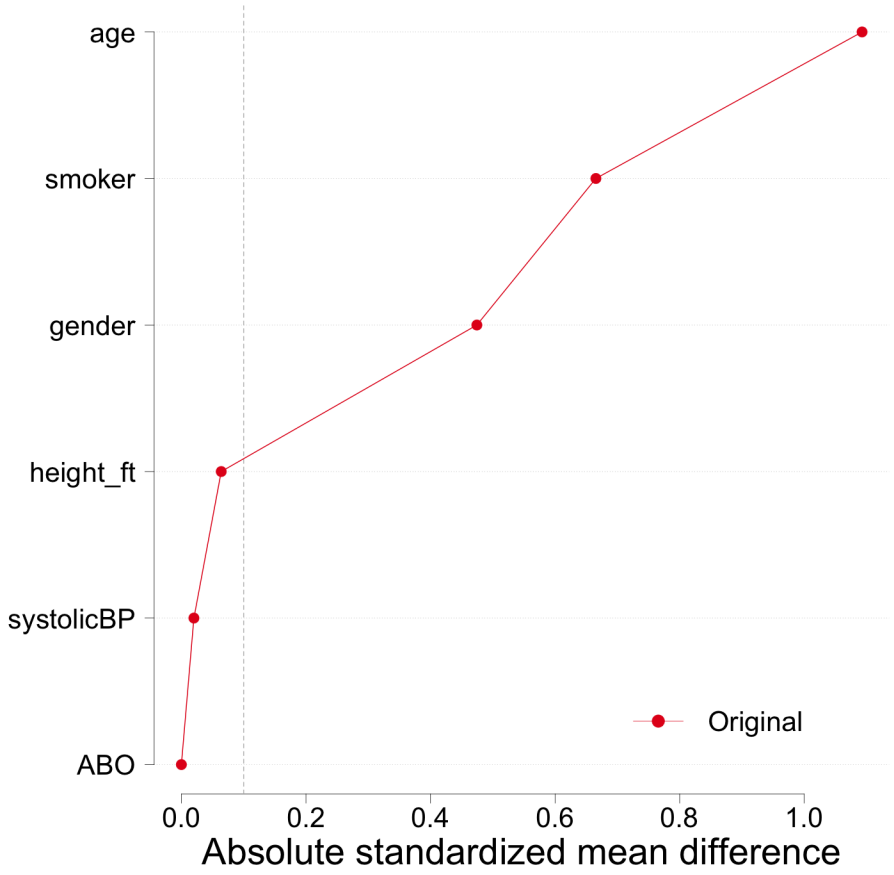

Figure 3.7: Absolute standardized mean difference (SMD) plot for the original sample, from the Compare tab. Because no pruning has been done and we have not chosen to display any of the optional weightings (checkboxes not shown), only the points for the original sample are shown in the plot. We have chosen to connect the points with a line (checkbox not shown); some analysts may prefer to view the plot without this option. As noted in Figures 3.3 and 3.6, there are large imbalances in `age` and `smoker`; `gender` is quite imbalanced as well. The absolute SMDs for these variables are greater than 0.1, the value marked on the plot by the dashed grey line and considered to be the highest acceptable level of imbalance by some researchers. Although the absolute SMDs for `height_ft`, `systolicBP`, and `ABO` are less than 0.1, it is important to note that the SMD plot gives no indication of the extreme lack of overlap in `height_ft` or of the informative missingness in `systolicBP`, both of which can be seen in the more detailed plots on the Prune tab (Figures 3.4 and 3.5).

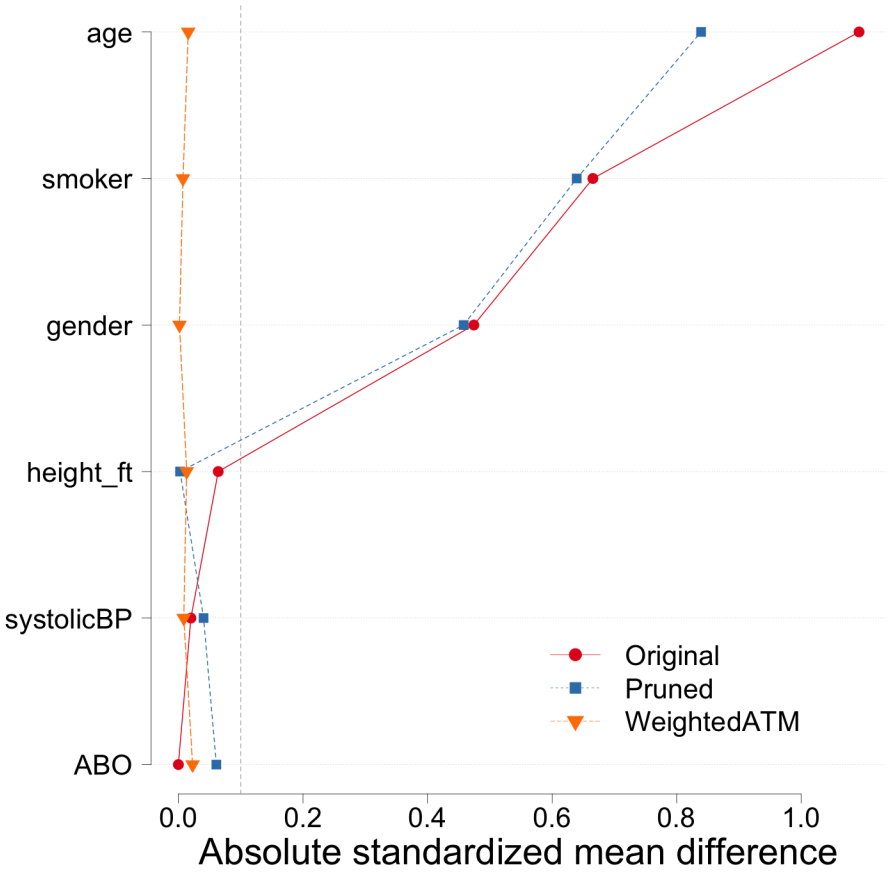

Figure 3.8: Absolute standardized mean difference (SMD) plot for the sample after updating the propensity score model and trying some initial pruning, from the Compare tab. After taking note of the informative missingness in `systolicBP` (see Figure 3.5), we returned to the Specify tab and updated the right-hand side of the propensity score model to include a missingness indicator for that variable: `age + height_ft + systolicBP + gender + smoker + ABO + is.na_systolicBP`. In addition, we pruned `age` to restrict the sample to subjects between the ages of 30 and 60, and we pruned `height_ft` to restrict the sample to subjects with heights in the range 5.25 ft–6 ft. This initial pruning reduced the sample to 224 exposed and 536 unexposed, for a total of 760 subjects. In addition to the points for the original sample (red circles, solid lines), the SMD plot now contains points for the pruned sample (blue squares, dashed line) and points for the ATM-weighted pruned sample (orange triangles, long-dashed line). Although we have chosen to connect the points from each sample with a line, some analysts may prefer to view the plot without this option (checkbox not shown). We see that the pruning improved the balance in terms of SMDs only slightly, suggesting that to get unbiased treatment effect estimates, we will want to follow the pruning with matching or weighting. We see that the ATM-weighted sample would have very good balance in terms of SMDs. Depending on the priorities for analysis, a next step might be loosening the restrictions on `age` or `height` to see whether we can still achieve good SMDs while keeping the sample a little bigger.

3.5 Discussion

With **Visual Pruner**, pruning decisions are informed by the propensity score distribution but not determined by it. Because the app allows iterative development of the propensity score model as well as easy re-estimation and reinspection of the estimated propensity scores after each pruning decision, it facilitates covariate-based cohort definition enhanced by extra information from the estimated propensity scores. As long as the analyst does not include the outcome variable in either the propensity score model or the set of variables to view and restrict, this iterative process is impervious to design bias from cherry-picking a cohort that yields a desired outcome. Observational studies in which the analyst has used **Visual Pruner** to assist with cohort selection can be both more convincing, because of the improved covariate balance and reduction in model dependence, and more generalizable, because of the simple univariate inclusion criteria.

In future versions of the app, the authors hope to add the ability to handle multiple treatment groups, as well as a default pruning feature implementing a modified version of one or more algorithmic pruning methods, for use as a starting place or comparison. In addition, while the app has benefitted greatly from the enhanced indexing and speed provided by the **data.table** package (Dowle et al., 2015), we would like to continue improving its speed with large datasets. We welcome feedback from researchers and analysts about other ways **Visual Pruner** could be improved.

3.6 Acknowledgements

The authors wish to thank Dale Plummer and the Department of Biostatistics, Vanderbilt University School of Medicine, for hosting **Visual Pruner**.

Portions of this paper first appeared in Poster Abstracts of IEEE VIS 2015. The authors are grateful to Meira Epplein, Bryan Shepherd, and Matt Shotwell for their comments and suggestions on the current manuscript. They and several other people, listed on the app's About tab, have also contributed valuable suggestions on the app itself.

Chapter 4

Bagged One-to-One Matching for Efficient and Robust Treatment Effect Estimation

4.1 Introduction

Observational studies, conducted in settings where random assignment to treatment (or exposure) groups is infeasible, present challenges due to bias from baseline confounders. One popular method for minimizing such bias is the estimation of effects within a matched cohort selected from the original study sample. In addition to reducing bias from observed confounders, matched cohorts are praised for being highly persuasive, reducing sensitivity to modeling choices, and offering some robustness to measurement error (Ho et al., 2007; Rosenbaum, 2010; Stuart, 2010; Waernbaum, 2012; Grijalva et al., 2015). However, the benefits of matching estimators can come at the cost of lower efficiency, due primarily to the exclusion of control (unexposed) subjects who possess potentially useful information. In studies of very large cohorts, researchers may gladly pay this cost in efficiency in exchange for the benefits of matching. However, in settings where researchers are reluctant to sacrifice power, an estimator that retains the central benefits of simple matching-based estimators while preserving efficiency would have great utility. In this paper we introduce the bagged one-to-one matching (BOOM) estimator, a highly efficient estimator that preserves all of the robustness and bias-minimizing benefits of simple matching estimators.

4.1.1 Treatment effect estimation after matching

In discussing the effect of a treatment on an outcome, we will use the framework and notation from Ho et al. (2007), which is based on the work of Neyman et al. (1935), Rubin (1974), and others. We limit ourselves to the case of a binary treatment or exposure and a continuous outcome. In this framework, subject i has two potential outcomes: $Y_i(1)$, the outcome if that subject receives the treatment of interest ($T_i = 1$) and $Y_i(0)$, the outcome if that subject receives the control condition ($T_i = 0$).[1] In general we are interested in estimating the average treatment effect, $\mathsf{E}\left[Y_i(1) - Y_i(0)\right]$, over all or some of the units in a sample or population.

[1] While the terms "treatment" and "control" reflect our goal of trying to make an observational study resemble a randomized trial as closely as possible, the terms "exposed" and "unexposed" are equally applicable, and readers with a background in epidemiology may prefer to use that terminology instead. We note in particular that we use the word "control" as it is used in randomized controlled trials, as opposed to the way it is used in case-control studies; that is, it is indicative of the absence of treatment, as opposed to the absence of the outcome of interest.

Matching is one of several techniques that have been developed to reduce confounding in the estimation of average treatment effects. Stuart (2010) and Ho et al. (2007) provide excellent overviews of the rationale for matching and the many options for the procedure itself. One commonly used version is one-to-one matching without replacement, in which each treated subject is matched to a unique control subject. If each treated subject cannot be matched closely enough with a unique control subject, analysts can use a caliper (Althauser and Rubin, 1970) in the matching process so that only some of the treated subjects are matched. Once the matched sample has been created, the estimation process can proceed with or without covariate adjustment, e.g. via regression.

In the absence of additional covariate adjustment, the estimate of the average treatment effect in the matched sample is simply the average of the paired differences in outcomes. For continuous Y and other collapsible outcomes (Greenland et al., 1999; Bishop et al., 1975), this is equal to the difference in the group means. Often, however, because the matched pairs are not matched exactly on each covariate, some residual bias remains, and further covariate adjustment is carried out through regression. If regression modeling is used within the matched sample, a variety of estimation approaches are possible, several of which are discussed in Schafer and Kang (2008) and Imbens (2004). One approach is to fit a multivariable regression model to the entire matched cohort, and to consider the treatment effect estimate for subject i to be the difference between subject i's predicted outcomes under treatment and control, or $\hat{Y}_i(1) - \hat{Y}_i(0)$. For collapsible models with no interactions involving the treatment, the estimate of the average treatment effect in the matched sample is equivalent to the treatment coefficient estimate from the fitted regression model.

None of the techniques discussed above can mitigate bias from unmeasured confounders. In the rest of this paper, we will assume that all possible confounders have been measured; this assumption is commonly referred to as ignorability, no omitted variable bias, exogeneity, or selection on observables (Ho et al., 2007; Imbens, 2004). We also make the Stable Unit Treatment Value Assumption (Rubin, 1980): we assume that there is only one version of the treatment and that one subject's treatment assignment has no effect on other subjects' outcomes. In addition, we focus on observational studies where subjects are selected on the basis of the treatment received, rather than on the basis of outcome; that is, we do not consider case-control designs.

4.1.2 Improved estimation via bagging

While one-to-one matching provides a transparent and compelling basis for estimation of treatment effects, it has the disadvantage of possibly discarding control subjects who could contribute useful information to the analysis. Several higher-efficiency alternatives to one-to-one matching have been proposed, each with its own strengths and weaknesses. These alternatives include 1:k matching with $k > 1$; stratification and weighting techniques such as full matching (Rosenbaum, 1991), Coarsened Exact Matching (Iacus et al., 2012), and inverse probability weighting (Rosenbaum, 1987); and regression-based covariate adjustment on the whole unweighted sample. Here we introduce an addition to this set of tools, using the technique of bagging, or bootstrap aggregating, developed by Breiman (1996): averaging estimates generated within a set of bootstrapped resamples taken from the original sample of interest in order to produce an improved estimate. Breiman advocated bagging in situations where "perturbing the learning set can cause significant changes in the predictor constructed" (Breiman, 1996). Because estimates resulting from one-to-one matching can, in some cases, depend to a great degree on which subjects are selected for inclusion, one-to-one matching is exactly the type of process that can be improved by bagging. The bagged one-to-one matching (BOOM) estimator uses bagging to improve upon the efficiency of estimation from one-to-one matching.

4.1.3 About this paper

The rest of the paper is organized as follows: Section 4.2 presents an overview of the BOOM estimator. Sections 4.3 and 4.4 present the methods for and results from a simulation study in which we investigate the performance of the BOOM estimator in a controlled situation and compare its performance to that of other commonly used estimators. Section 4.5 describes a brief case study. We discuss the findings from the simulation study and the case study in Section 4.6.

4.2 Methods: The bagged one-to-one matching estimator

Here we present an overview of the process for obtaining point estimates and standard error estimates from bagged one-to-one matching. R functions for implementing these steps can be found at https://github.com/LaurenSamuels/BOOM.

4.2.1 The estimator itself

Although the bagged one-to-one matching (BOOM) estimator uses the general bagging methodology introduced by Breiman (1996), our specific approach is inspired by the "complex bootstrap" of Austin and Small (2014). While Austin and Small investigated the complex bootstrap as a way to obtain standard error estimates for a simple matching-based estimator, we modify the process to obtain an estimate of treatment effect.

The BOOM process can take many forms. In broad terms, the process consists of six steps, two of which are optional:

1. (optional) Prune the original dataset to remove regions of extreme nonoverlap between groups.

2. Develop the general form of a distance measure (e.g., propensity score model) on the whole (or pruned) sample.

3. Choose a one-to-one matching algorithm.

4. (optional) Choose the general form of an outcome model or modeling approach (covariate adjustment).

5. Resample the data B times, and obtain a treatment effect estimate in each of the B bootstrap resamples, using the general distance measure from Step 2, the matching algorithm from Step 3, and, if desired, the outcome-modeling approach from Step 4.

6. Take the average of the B treatment effect estimates.

We will now go into more detail about the six steps:

4.2.1.1 Step 1 (optional): Pruning

We begin with N_{t0} treated subjects and N_{c0} control subjects. In many cases, the distributions of the covariates in the two groups will have areas of obvious nonoverlap. As a first step, "pruning" the data— restricting the sample in order to eliminate these regions of obvious nonoverlap— can reduce model dependence (Ho et al., 2007). Although one-to-one matching can also be seen as a type of pruning, here we draw a distinction with pruning before matching, particularly because if propensity scores are being used as the distance measure in Step 2, the performance of any proposed propensity score model can be assessed only in regions of the data where common

support exists, and so pruning is an essential step before the development of a reliable propensity score model (King and Zeng, 2007; Ho et al., 2007). After pruning, we have N_t treated subjects and N_c control subjects, for a total of N subjects, and our inference about treatment effects will apply to this pruned sample.

4.2.1.2 Step 2: Development of distance measure

The next step is to decide on a distance measure for use in the matching process. Commonly used distance measures include propensity scores (Rosenbaum and Rubin, 1983), prognostic scores (Hansen, 2008), and reweighted or generalized Mahalanobis distance (Greevy et al., 2012; Diamond and Sekhon, 2013). Although the distances themselves will be calculated within each bootstrap sample in Step 5, the analyst will fix some element(s) of the calculation process here in Step 2. For example, an analyst who wants to match on propensity scores and to estimate those propensity scores using logistic regression will develop the general form of the propensity score model here in Step 2, and then fit that same general model within each bootstrap sample in Step 5.

4.2.1.3 Step 3: Selection of matching algorithm

BOOM uses one-to-one matching without replacement. Algorithms for one-to-one matching without replacement have several options, many of which are discussed in Stuart (2010); in this step the analyst must choose the type of algorithm (optimal or greedy), and, if a greedy algorithm is chosen, the order in which to process the subjects. A further decision is whether to use a caliper in the matching process.

4.2.1.4 Step 4 (optional): Outcome model

As mentioned in Section 4.1.1, treatment effect estimates can be obtained with or without further covariate adjustment after matching. If the analyst does not want to do any further covariate adjustment, there is nothing to do in this step. If the analyst does want to do further covariate adjustment, this step consists of specifying the general form of the outcome model. Although the current version of the BOOM software limits covariate adjustment to linear models with no treatment interactions, other types of outcome models could certainly be integrated.

4.2.1.5 Step 5: Obtaining bootstrap estimates

Step 5 consists of conducting steps 5a–5d B times, which means that the analyst must first choose B, the number of bootstrap resamples. The larger B is, the

less Monte Carlo error there will be in the BOOM estimator and, in particular, in estimates of its standard error (discussed below in Section 4.2.2). To get reliable standard error estimates, Efron (2014) suggests a method for choosing B using calculations based on the jackknife, and Wager et al. (2014) state that for many bagged estimators, B needs to be $\Theta(n^{1.5})$ unless a bias-corrected version of the standard error estimate is used, in which case B can be $\Theta(n)$. It is unclear whether BOOM falls into this class of estimators; in addition, for any given sample size, the ideal B will depend on the variability of the point estimate from resample to resample. In our simulations (presented below), we found adequate, but not perfect, performance of the bias-corrected standard error estimates using $B = 10n$. We recommend that researchers working with a single dataset rather than a series of simulations begin by either using Efron's jackknife-based calculations or by setting B to several times the sample size, and increasing B as feasible from that point until the differences between the bias-corrected and uncorrected standard error estimates are acceptable in the context of the study.

Step 5a: Resample $N_t + N_c = N$ subjects In a departure from Austin and Small's complex bootstrap (Austin and Small, 2014), resampling in BOOM conditions on the original sizes of the treatment and control group, following the advice of Hesterberg (2014). That is, rather than resampling N subjects without regard to treatment group, we resample N_t subjects from the treatment group and N_c subjects from the control group in order to condition on the observed information from the original dataset.

Step 5b: Generate distance measures on the bootstrap sample Here we generate the distance measures using the method developed in Step 2. For example, if using propensity scores estimated via logistic regression, we would keep the general model from Step 2, but fit the model to this particular bootstrap sample.

Step 5c: Create a matched sample In this step we use the matching algorithm selected in Step 3 and the resample-specific distance measures from Step 5b to create a matched sample. Depending on whether a caliper is specified (and exceeded) or not, this step will select all or some of the treated subjects in the bootstrap sample and match each to a single control subject. Note that a unique treated subject from the original sample could appear in the bootstrap sample more than once, and then also enter into the matched sample more than once, with each appearance matched to a different control subject, or even to

the same control subject if that control subject also appears in the bootstrap sample more than once.

Step 5d: Estimate and record the treatment effect in the matched sample As discussed in Section 4.1.1, there are several ways to estimate the treatment effect in a matched sample. If in Step 4, the analyst decided not to do further covariate adjustment after matching, the treatment effect estimate is simply the mean of the differences in outcome for the matched pairs, or equivalently the difference in group means for the two groups. On the other hand, if in Step 4, the analyst specified an outcome model, in this step the model will be fit to the matched sample, and the treatment effect estimate obtained using either the coefficient estimate for the treatment term (if no treatment interactions are used) or one of the methods described in Schafer and Kang (2008) and Imbens (2004).

4.2.1.6 Step 6: Aggregate to get the bagged estimate

The BOOM estimate is simply the average of the B treatment effect estimates. In most cases, the analyst will also want to generate estimates of the standard error and confidence intervals in this step; more information follows.

4.2.2 Estimates for the standard error of the BOOM estimator

In most statistical applications, an estimator is useful only if one has a way to describe the uncertainty in the estimate it produces. It is tempting to simply use the standard deviation of the B treatment effect estimates as the estimate of the standard error of the bagged estimator, but Efron (2014) shows that this approach, while easy to implement, is a conservative one.

Sexton and Laake (2009) and Efron (2014) discuss several other ways of obtaining standard error estimates for bagged estimators. Both papers discuss a "brute force" approach which, in the case of BOOM, would consist of taking B' bootstrap resamples from the original (pruned) sample and conducting the BOOM estimation process in each one, and then taking the standard deviation of the B' BOOM estimates. The process could be implemented as a second-level bootstrap (letting $B' = B$ and using the B resamples used to create the main BOOM estimate as the starting point for the next level). Both papers argue that, while the brute-force approach is theoretically strong, it is too computationally intensive to be practical. Sexton and Laake offer variations on the brute-force estimator ("Biased Bootstrap," "Noisy Bootstrap," and

jackknife after bagging), all of which involve some degree of additional resampling, and thus still some amount of intensive computation.

Efron (2014) offers an approach that uses only the first-level bootstrap estimates and does not require additional bootstrapping; we have chosen this method to use with the BOOM estimator. Efron's approach, illustrated with slight variation in Wager et al. (2014), is the nonparametric delta-method estimate and is based on Efron's earlier work with the influence function and infinitesimal jackknife estimates of standard error. For finite B, Efron's approximation to the nonparametric delta method estimate of the variance of a bagged estimator is as follows:

$$\widetilde{\mathrm{sd}}_B^2 = \sum_{i=1}^{n} \widehat{\mathrm{cov}}_i^2, \tag{4.1}$$

where

$$\widehat{\mathrm{cov}}_i = \sum_{j=1}^{B} (C_{ji}^* - C_{\cdot i}^*)(t_j^* - t_\cdot^*)/B, \tag{4.2}$$

with C_{ji}^* the number of times subject i appears in bootstrap sample j, and t_j^* the estimate of interest from bootstrap sample j.[2]

The estimate in Eq. 4.1, because it relies on a finite number of bootstrap samples, is an approximation to the true nonparametric delta-method variance estimate $\widetilde{\mathrm{sd}}^2$. The finite-$B$ approximation is always biased upwards, and both Efron (2014) and Wager et al. (2014) offer bias-corrected versions. As Wager et al. (2014) point out, the amount of bias in the standard error estimate depends on the variability of the underlying estimation procedure in the given dataset. Efron's bias-corrected estimate is

$$\widetilde{\mathrm{sd}}_{B,BC}^2 = \widetilde{\mathrm{sd}}_B^2 - \frac{1}{B^2} \sum_{i=1}^{n} \sum_{j=1}^{B} (Z_{ji}^* - \widehat{\mathrm{cov}}_i)^2, \tag{4.3}$$

where

$$Z_{ji}^* = (C_{ji}^* - 1)(t_j^* - t_\cdot^*), \tag{4.4}$$

where we have added the "BC" notation and are substituting $t_\cdot^*$ for s_0, the expected value of t_j^*.

Following Efron (2014), we use the estimates $\widetilde{\mathrm{sd}}_{B(,BC)}$ in the construction of approximate 95% Wald confidence intervals for use with the BOOM estimator,

$$(\hat{\theta}_B - 1.96\,\widetilde{\mathrm{sd}}_{B(,BC)},\ \hat{\theta}_B + 1.96\,\widetilde{\mathrm{sd}}_{B(,BC)}),$$

[2]One typographical error from the original paper has been corrected in Eq. 4.2. We have also changed the notation slightly for compatibility with notation in the remainder of the present paper.

where $\hat{\theta}_B$ denotes the BOOM estimate. Efron also discusses other approaches to the construction of confidence intervals for bagged estimators, some of which may be appropriate for use with BOOM.

4.2.2.1 Using Efron's estimates with two-group resampling

As mentioned in Section 4.2.1.5, in BOOM estimation we resample separately from the treatment and control groups in each iteration. Efron's standard-error approximation (Eq. 4.1) is derived under the assumption that single-group resampling was used and that for each bootstrap sample j, the vector of counts $\mathbf{C}^*_\mathbf{j} = (C^*_{j1}, C^*_{j2}, \ldots, C^*_{jn})$ follows a multinomial distribution. Thus Efron's standard-error estimator is not perfectly compatible with the BOOM estimation process.

In situations where the outcome of interest is a difference in means and the analyst does not want to do further covariate adjustment after the matching within each bootstrap sample, the BOOM estimation process can be seen as two separate estimation processes, each of which supports the assumptions of Efron's standard error estimate. That is, because the BOOM estimate can be seen as an average difference in group means, we can calculate Efron's estimated variance separately for the average outcomes in the treatment group and the control group, then add the variances to get the estimated variance of the BOOM estimate. If, however, the analyst wants to do covariate adjustment via regression within each matched bootstrap sample and to use the average treatment coefficient estimate as the basis of the BOOM estimate, the BOOM estimate cannot be seen as a function of two separate estimation processes, and so there is a slight violation of the assumptions underlying Efron's standard error estimator. We conducted simulations (not presented here) that suggest that the violation is only slight and makes little difference in terms of the resulting standard error estimate.

4.2.3 Modifying the BOOM procedure to incorporate estimation of the standard error

Using Efron's method to obtain standard error estimates for the BOOM estimator requires a few additions to the estimation process, mainly in Step 5 (Section 4.2.1.5). In Step 5a, the vector of counts should be saved for use in Eq. 4.2; if no covariate adjustment will be done after matching, the count vectors should be saved separately for each group. Note that these counts are the number of times each subject appears in the bootstrap sample, regardless of how many times that subject

appears in the subsequently created matched sample. In Step 5d, if no further covariate adjustment is done, the mean for each group should be saved. Then, in a final Step 7, Efron's estimates for standard errors and confidence intervals (original and bias-corrected) can be calculated using the B sets of count vectors and group means or effect estimates. These modifications are incorporated into the code at https://github.com/LaurenSamuels/BOOM.

4.2.4 Further variations

Many other modifications to the BOOM procedure are possible. In Section 4.2 we discussed fixing some elements of the distance-generation process in Step 2, and then varying other elements within each bootstrap sample in Step 5. These portions of the algorithm could be adjusted, however, to allow for more or less "wiggle" in the bootstrap estimates. For example, an analyst using propensity scores estimated via logistic regression as the distance measure could, rather than fixing the general propensity score model in Step 2, specify a set of possible model terms in Step 2, and then each bootstrap sample in Step 5 could use a propensity score model that uses a random subset of those terms, similar to the approach used in a random forest (Breiman, 2001). Alternatively, to prevent the estimated propensity scores from varying across bootstrap samples, instead of specifying only the general form of the propensity score model in Step 2, the analyst could fit the model on the whole (pruned) sample in Step 2, and use those fixed scores in each bootstrap sample in Step 5.

The matching algorithm could also be varied within the BOOM process, rather than being fixed ahead of time in Step 3. For example, greedy matching could be used in some resamples, with optimal matching used in others. We do not recommend, however, varying the use of a caliper from resample to resample, because this choice is tied to the choice of estimand.

Similarly, the outcome model could be varied within the BOOM process, perhaps in the manner described in (Efron, 2014), rather than being fixed in Step 4. In fact, if the model specified in Step 4 was not completely prespecified, the standard errors from the BOOM procedure may be more accurate if the model selection process is repeated in each of the BOOM iterations; the question of standard error estimation after model selection was the motivation for Efron's paper.

4.2.5 BOOM weights: Describing the BOOM cohort

In many disciplines it is traditional for the first table in a published article to present descriptive statistics by treatment group; by presenting a well balanced table, analysts can make a convincing case that the groups being compared are indeed comparable. One-to-one matching without replacement lends itself to the straightforward creation of a "Table 1." Because the BOOM process averages results over many matched cohorts, the exact cohort being studied is not instantly apparent as it is with one-to-one matching. The BOOM process, however, can also be seen as a weight-generating process: each subject has a weight equal to the total number of times that subject appears in a matched cohort divided by the total number of times the subject appears in a bootstrap resample. Thus the minimum possible BOOM weight is zero and the maximum possible is one. (If a subject never appears in a bootstrap resample, we assign that subject a weight of zero; we recommend, however, that B be set high enough that the chance of this happening is extremely small.) A weighted "Table 1" can be created using the BOOM weights, using software such as the R packages survey (Lumley, 2014, 2004) and tableone (Yoshida and Bohn, 2015). We explore the BOOM weights and their properties in greater detail in a forthcoming manuscript (Samuels and Greevy, Jr., 2016*b*).

4.3 Simulation study: Methods

To investigate the properties of the BOOM estimator in a controlled setting, we conducted a simulation study. In each of 5,000 iterations, we generated data under three different treatment prevalences, then estimated the average treatment effect using both the new estimator and three techniques that are currently in use. We investigated the performance of BOOM and the other techniques in a total of nine different scenarios that varied according to the presence and correctness of the treatment model and the outcome model. We compared the techniques on mean squared error, bias, variance, accuracy of standard error estimation, and coverage of nominal 95% confidence intervals. We conducted the simulation study using the R statistical software package (R Core Team, 2015). Code used for the study can be found in Appendix A (Section 4.8).

4.3.1 Data generation

To generate data for the simulation study, we began with the data-generation process used by Austin and Small (2014) and modified it based on the procedures in

Setoguchi et al. (2008) and Lee et al. (2010). Specific details follow.

4.3.1.1 Covariates

We reduced the sample size from Austin and Small (2014) to $N = 1000$, but otherwise followed their covariate-generation process exactly in each iteration, producing a covariate matrix with 10 uncorrelated standard normal variables, X_1 through X_{10}.

4.3.1.2 Treatment assignment

In each iteration, the covariate matrix formed the core of three datasets, each with a different level of treatment prevalence: 5%, 10%, or 20% treated on average. Because of the complexity of the experimental design, in the remainder of this paper we will focus on the simulations with 10% average treatment prevalence. As in Austin and Small (2014), we assigned subjects to treatment with probability p, with p a function of variables X_1 through X_7. Because a common manner of misspecification of propensity score models is the omission of squared terms and interactions, and because we wanted to allow for this type of misspecification in our simulations, we modified Austin and Small's assignment mechanism in order to include squared terms and an interaction, in the spirit of Setoguchi et al. (2008) and Lee et al. (2010). In particular, each subject was assigned to the treatment group with probability p_i, where

$$\begin{aligned}\text{logit}(p_i) = \beta_0 + \beta_L x_{1i} + \beta_M x_{2i} + \beta_H x_{3i} + 0.5\beta_L x_{4i} + 0.3\beta_L x_{4i}^2 \\ + 0.5\beta_M x_{5i} + 0.5\beta_H x_{6i} + 0.3\beta_H x_{5i} x_{6i} + 0.5\beta_{VH} x_{7i} + 0.3\beta_{VH} x_{7i}^2 \,.\end{aligned} \tag{4.5}$$

The intercept β_0 was tuned to produce an average treatment prevalence of 10%, and the subscripts on the other coefficients indicate a Low, Medium, High, or Very High value. The left panel of Figure 4.1 shows the relationship between covariate X_7 and treatment assignment in a sample dataset generated using this mechanism.

4.3.1.3 Outcome

Following the general approach in Austin and Small (2014), we then used variables X_4 through X_{10} and the treatment indicator to generate an outcome for each subject. As with the treatment assignment, we modified Austin and Small's method to include squared terms and interactions, to allow for later omission of these terms in misspecified outcome models. The outcome Y for each subject was calculated as

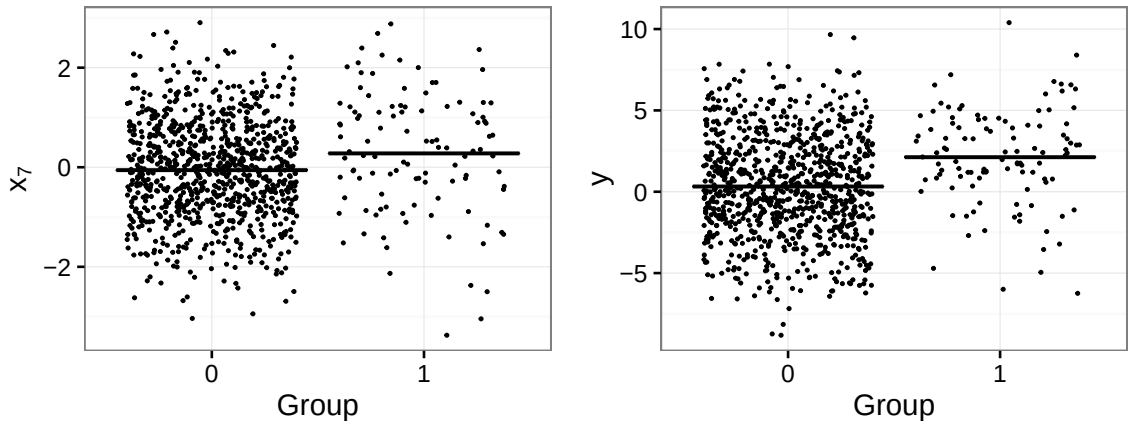

Figure 4.1: Two illustrations from example dataset from simulation study ($N = 1000$, with 107 in Group 1 (Treated)). In each plot, points are jittered horizontally to minimize overplotting, and horizontal lines are group means. **Left:** Distribution of covariate x_7 by treatment group. **Right:** Distribution of outcome (y) by treatment group, showing the bias in the unadjusted difference in means: the true treatment effect is 1, but unadjusted difference in group means is 1.8.

follows:

$$\begin{aligned} Y_i = T_i &+ 0.5\beta_L x_{4i} + 0.3\beta_L x_{4i}^2 + 0.5\beta_M x_{5i} + 0.5\beta_H x_{6i} + 0.3\beta_H x_{5i} x_{6i} \\ &+ 0.5\beta_{VH} x_{7i} + 0.3\beta_{VH} x_{7i}^2 + \beta_L x_{8i} + \beta_M x_{9i} + \beta_H x_{10i} + \epsilon_i \,, \end{aligned} \tag{4.6}$$

where T_i is the treatment indicator, $\epsilon_i \sim N(0, \sigma = 3)$, and, as before, the subscripts on the coefficients indicate a Low, Medium, High, or Very High value. Thus the true treatment effect in the simulation study is 1.

The right panel of Figure 4.1 shows the relationship between treatment group and outcome in a sample dataset. In this dataset, the raw difference in group means for the outcome is 1.8 rather than the true treatment effect of 1, illustrating the bias that has arisen from group differences in the distributions of covariates related to both treatment assignment and outcome, as is typical in an observational study.

4.3.2 The particular implementation of BOOM

Section 4.2 gives an overview of the BOOM estimation process in general; here we provide details about the particular implementation used in the simulation study. We did not prune the original datasets first (Step 1, Section 4.2.1.1), both because the data-generation process produced only mild nonoverlap and because of the impracticality of making individual pruning decisions on 5,000 simulated datasets. We used propensity scores as our distance measure (Step 2, Section 4.2.1.2) and estimated the

propensity scores using logistic regression with the lrm() function in the rms package (Harrell Jr., 2015). To create the matched samples (Step 3, Section 4.2.1.3), we used the Matching package (Sekhon, 2011) to conduct greedy matching on the logit of the estimated propensity scores, processing the treated subjects in random order, with ties broken at random. Within the matching process we chose to use a caliper to further reduce bias. We used a caliper of 0.2 times the standard deviation of the logit propensity scores, following Austin and Small (2014) and common practice. In scenarios in which we did further covariate adjustment (Step 4, Section 4.2.1.4), we fit a multiple linear regression using R's lm() function. For each BOOM estimate, we let $B = 10{,}000$ (Step 5, Section 4.2.1.5); that is, we aggregated estimates from 10,000 bootstrap resamples.

4.3.3 The six scenarios (model combinations)

In each iteration, we estimated the average treatment effect using the BOOM estimator under six scenarios: under the ideal circumstance in which the analyst knows both the right propensity score model and the right outcome model, and under five more realistic circumstances, with the most realistic following Kang and Schafer (2007) in trying to achieve "a simulated example where both models are incorrect but neither is grossly misspecified." In total we examined the six combinations of two propensity score models ("Right" and "Wrong") and three outcome models ("None," "Right," and "Wrong").

For the "Right" propensity score model, we followed Austin and Small (2014) and the currently accepted best practice: instead of using the true treatment-assignment model (Eq. 4.5), we used all covariates that affected outcome. That is, our "Right" propensity score model is

$$\begin{aligned}\text{logit}(p_i) = \beta_0 &+ \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \beta_4 x_{4i} + \beta_5 x_{4i}^2 \\ &+ \beta_6 x_{5i} + \beta_7 x_{6i} + \beta_8 x_{5i} x_{6i} + \beta_9 x_{7i} + \beta_{10} x_{7i}^2 + \beta_{11} x_{8i} + \beta_{12} x_{9i} + \beta_{13} x_{10i}.\end{aligned} \tag{4.7}$$

Our "Wrong" propensity score model is identical to Eq. 4.7, except that the two squared terms and the interaction are omitted. This misspecification leads to generally small differences in the estimated propensity scores, as shown in the top panel of Figure 4.2.

For the outcome models, "None" uses no covariate adjustment beyond the matching (i.e., a difference in group means in Step 5d (Section 4.2.1.5)). While the estimates

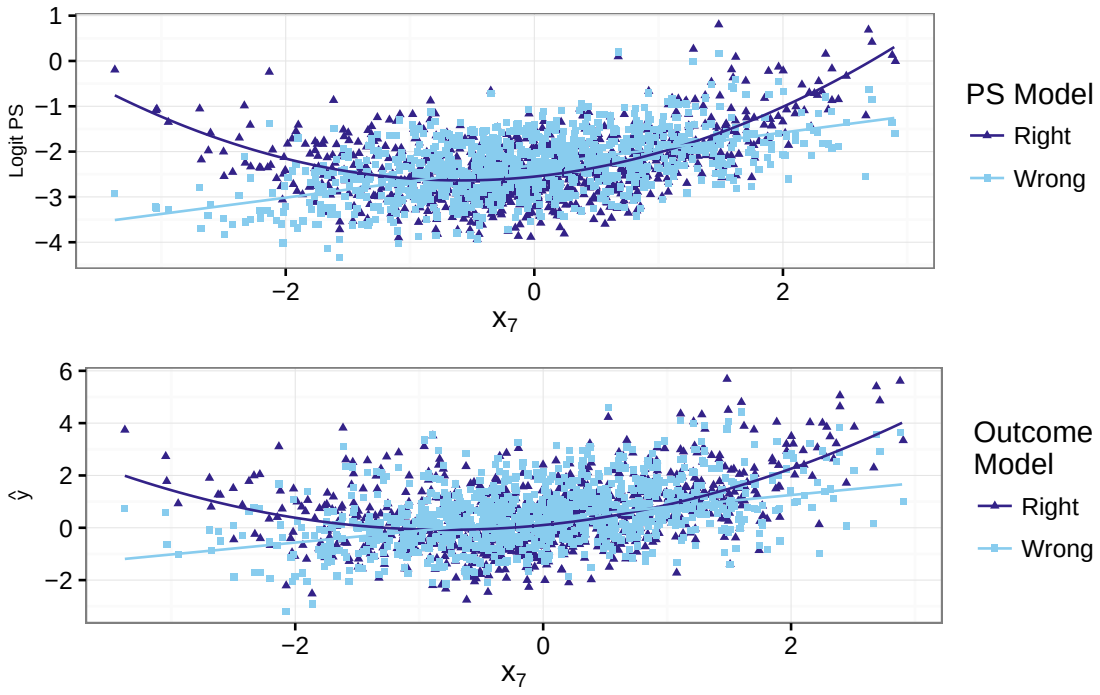

Figure 4.2: Top: Estimated logit propensity scores for our example dataset under the "Right" and "Wrong" models, by covariate x_7. Bottom: Fitted outcomes for our example dataset under the "Right" and "Wrong" models, by covariate x_7.

that do not use outcome models are marginal estimates, we avoid using this terminology because in the causal inference framework used here, even the estimates that use additional covariate adjustment are marginal estimates, as they are averages taken over the whole group (Section 4.1.1).

The "Right" outcome model is indeed the right outcome model,

$$\begin{aligned} Y_i = T_i + \beta_L x_{4i} + \beta_L x_{4i}^2 + \beta_M x_{5i} + \beta_H x_{6i} + \beta_H x_{5i} x_{6i} \\ + \beta_{VH} x_{7i} + \beta_{VH} x_{7i}^2 \beta_L x_{8i} + \beta_M x_{9i} + \beta_H x_{10i} + \epsilon_i, \end{aligned} \tag{4.8}$$

where T_i is the treatment indicator and ϵ_i is Normally distributed with mean zero. Our "Wrong" outcome model is identical to Eq. 4.8, except that the two squared terms and the interaction are omitted. This misspecification leads to generally small changes in the fitted outcome values, as shown in the bottom panel of Figure 4.2.

4.3.4 Comparison to other methods

In order to get a sense of how the BOOM estimator performs in comparison to other commonly used methods, we first conducted three variations of ordinary least

Table 4.1: Experimental design for simulation study. The 21 estimates shown below are calculated in each of 5,000 iterations. PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability weighting.

	Outcome model		
PS Model	None	Right	Wrong
None	OLS	OLS	OLS
Right	BOOM OOM IPW	BOOM OOM IPW	BOOM OOM IPW
Wrong	BOOM OOM IPW	BOOM OOM IPW	BOOM OOM IPW

squares estimation on the whole (unmatched) sample, to obtain benchmarks. We then estimated average treatment effects using two other commonly used methods, one-to-one matching and inverse probability weighting, under the six scenarios in which we examined BOOM. In all, this gave us 15 estimates in addition to the six BOOM estimates, for a total of 21 estimates per iteration; Table 4.1 summarizes the design. The methods used for comparison are described in greater detail below.

4.3.4.1 Ordinary least squares (OLS)

In each iteration we calculated three ordinary least squares (OLS) estimates on the whole sample, without any matching or weighting, to use as benchmarks. To illustrate the degree of confounding in the sample, we calculated a simple difference of means without adjusting for any of the covariates. Then, to obtain a "gold standard" estimate, we also fit a linear regression using the true outcome model. It is, of course, unlikely that a researcher would be naive enough to use the first approach, or lucky enough to be able to use the second, but we wanted these estimates to serve as bookends for our other estimates. We also included a more realistic scenario, in which we fit a linear regression using the misspecified outcome model discussed in Section 4.3.3. To get standard error estimates and 95% confidence intervals for the OLS approaches, we used an unequal-variance t-test for the unadjusted scenario, and R's lm() and confint() functions for the scenarios with covariate adjustment.

4.3.4.2 One-to-one matching (OOM)

We obtained estimates using one-to-one matching (OOM) in the same six scenarios under which we evaluated BOOM: all combinations of the two propensity score models ("Right" and "Wrong") and the three outcome models ("None," "Right," and "Wrong"). The one-to-one matching and estimation procedure we used is exactly the same as the matching and estimation procedure used within BOOM in this study (see Section 4.3.2), except that we did it just once. To obtain standard error estimates and 95% confidence intervals for the OOM estimators, under the scenarios with no outcome model we used a paired t-test as recommended by Austin and Small (2014), and under the scenarios using the Right and Wrong outcome models, we used R's lm() and confint() functions.

4.3.4.3 Inverse probability weighting (IPW)

For a final set of comparisons under the six scenarios, we also generated inverse probability weighting (IPW) estimates. We used the svydesign() and svyglm() functions from the survey package (Lumley, 2014, 2004) to obtain treatment effect and standard error estimates and Wald 95% confidence intervals in all six scenarios. In the IPW estimation we used weights $w_i = T_i + (1 - T_i)\,\hat{e}_i/(1 - \hat{e}_i)$, where T_i is the treatment indicator and $\hat{e}_i$ is the estimated propensity score, to estimate the average treatment effect on the treated units, or ATT (Austin, 2013).

4.3.5 Fairness of the comparisons

As noted in Section 4.1.1, methods for observational studies usually attempt to estimate the average treatment effect over all or some of the units in a sample or population. Depending on the sample, different techniques can estimate the average treatment effect over all units (ATE), the average treatment effect on the treated units (ATT), the average treatment effect on the evenly matchable units (ATM; see Samuels and Greevy, Jr. (2016*b*)), and a variety of other estimands. Thus in any comparison of methods, it is important to take into account the quantities the methods are actually estimating, in order to ensure that the comparisons are fair. In this simulation study, the OLS methods estimate the ATE, the OOM and BOOM methods estimate the ATM, and the IPW methods estimate the ATT. Because of the collapsible outcome and the constant treatment effect, however, the three estimands are identical, and thus the comparisons are fair in this setting.

4.4 Simulation study: Results

In this section we present the mean squared error, bias, variance, accuracy of standard error estimates, and coverage of nominal 95% confidence intervals from the simulation study. Overall, in our simulations BOOM performed as well as one-to-one matching (OOM) in terms of bias, far better than OOM in terms of mean squared error and variance, and worse than OOM but still adequately in terms of accuracy of standard error estimates and coverage rates.

4.4.1 Mean squared error

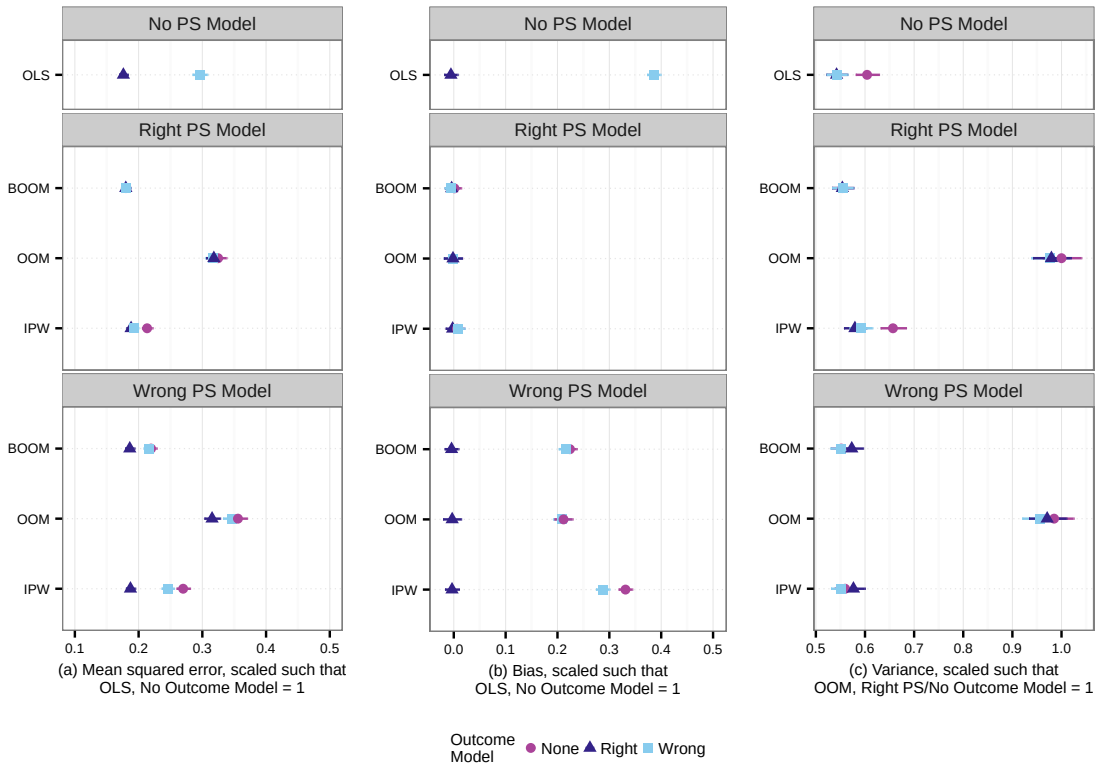

Figure 4.3: (a) Mean squared error, (b) bias, and (c) variance from simulation study. 95% confidence intervals are shown in all three panels. PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability weighting.

Figure 4.3(a) shows the mean squared error (MSE) from the estimators in the nine scenarios. The values shown in Figure 4.3(a) have been scaled by the highest MSE value, that of the OLS estimator in the "No PS Model, No Outcome Model" scenario, to simplify interpretation. As expected, the OLS estimator from the "No PS Model, Right Outcome Model" scenario yielded the lowest MSE in our study; we included

this scenario in the study to provide a benchmark for the best possible performance. BOOM using the "Right" propensity score model did almost as well as this gold standard, regardless of the outcome model used. That is, with a good propensity score model, adjusting correctly for covariates resulted in no further improvement to the BOOM estimator's MSE, and adjusting incorrectly did no harm. Using this propensity score model, BOOM performed better in terms of MSE than one-to-one matching (OOM), with MSE between 55% and 57% of the OOM MSE. The BOOM estimator also performed better than using OLS with the whole sample and a slightly misspecified outcome model. The IPW estimator performed similarly to the BOOM estimator when the "Right" propensity score model was used.

With a slightly misspecified propensity score model and the right outcome model, BOOM still did nearly as well as using OLS with the whole sample and the true outcome model in terms of MSE. As with the "Right" propensity score model, BOOM under a slightly misspecified propensity score model outperformed both the whole-sample OLS estimator with the misspecified outcome model and OOM, even when the BOOM estimator had an absent or incorrect outcome model. The IPW estimator performed similarly to the BOOM estimator here as well.

4.4.2 Bias

Figure 4.3(b) shows the bias from the estimators in the nine scenarios, scaled by the value from the most biased estimator, the OLS estimator in the "No PS Model, No Outcome Model" scenario. As with MSE, the OLS estimator in the "No PS Model, Right Outcome Model" scenario serves as our gold standard here and is, of course, completely unbiased. As one might hope, the other three estimators (BOOM, OOM, and IPW) were also unbiased when the correct outcome model was used, regardless of the propensity score model used. All three were also unbiased when a good propensity score model was used, regardless of the subsequent covariate adjustment (or lack thereof), thus exhibiting what Waernbaum (2012) calls "finite sample robustness" if not true double robustness. Under a misspecified propensity score model with no or misspecified outcome model, both BOOM and OOM were less biased than the whole-sample OLS estimator with the misspecified outcome model. This finding was true for the IPW estimators to a lesser degree. BOOM provided no improvement over OOM in terms of bias; this is as expected, since the advantage of BOOM is in variance reduction.

4.4.3 Variance

Figure 4.3(c) shows the variance of the estimators in the nine scenarios. Here we have once again scaled by the highest value, but in contrast to MSE and bias, where the highest value came from the OLS estimator in the "No PS Model, No Outcome Model" scenario, the highest variance arose from the one-to-one matching (OOM) estimator in the "Right PS Model, No Outcome Model" scenario. Indeed, the potential for increased variance when using one-to-one matching was a key motivator for the development of BOOM, and BOOM did provide a large improvement, with variances ranging from 55% to 57% of the variance of the OOM estimators under a good propensity score model. The estimators with the lowest variance were the OLS estimators using either correct or slightly misspecified covariate adjustment, and under a good propensity score model, the variance of the BOOM estimator was almost as low as these, regardless of the outcome model used. In these scenarios the IPW estimator performed similarly to the BOOM estimator, with slightly higher variance when no further covariate adjustment was used. Under a misspecified propensity score model, both BOOM and IPW performed almost as well as the covariate-adjusted OLS estimators, and far better than OOM, regardless of outcome model.

4.4.4 Accuracy of standard error estimates

Figure 4.4(a) explores the accuracy of the standard error estimates for the estimators in the simulation study. The shaded rectangles in the figure show the 95% confidence intervals for the true standard error of the treatment effect estimates; each rectangle is centered at the actual standard deviation of the 5,000 treatment effect estimates for that estimator in that scenario, and the width of the rectangles is due to Monte Carlo uncertainty about the true standard error. The plotting symbols in Figure 4.4(a) mark the mean standard error estimates from each particular scenario and method. Plotting symbols that are far from their corresponding shaded rectangles suggest that the standard-error estimation process may be flawed for that estimator in at least that particular scenario.

For context, we present the results for the established methods— OLS, OOM, and IPW— first. The whole-sample OLS methods show good performance; the estimated standard errors are very close to the empirical "truth." The regression-based standard error estimates for the OOM estimators using covariate adjustment ("Right" or "Wrong" outcome model) also perform very well, regardless of the propensity score specification. For the OOM estimators that did not use further covariate adjustment,

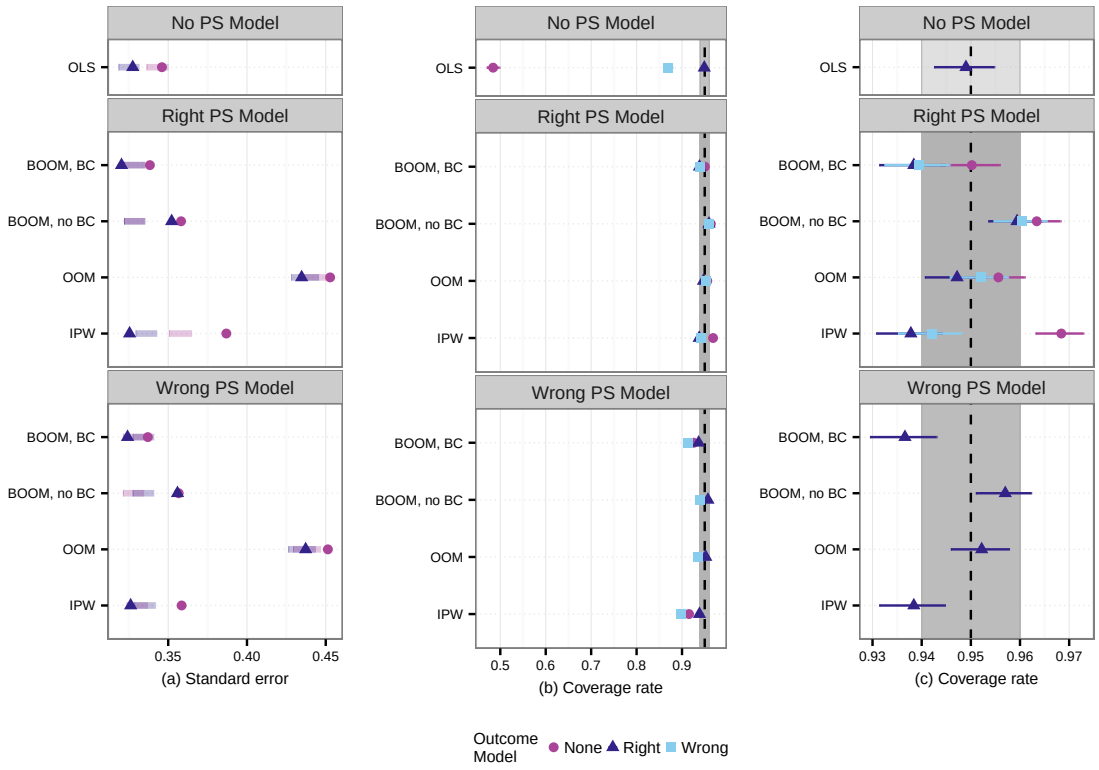

Figure 4.4: (a) Mean standard error (SE) estimates from simulation study; 95% confidence intervals for these estimates are narrower than the plotting symbols. 95% confidence interval for true SE in each case (actual standard deviation of the treatment effect estimates) is shown as a shaded rectangle behind the plotting symbols; confidence interval for true SE is due to Monte Carlo uncertainty. Results for "Wrong" outcome model are very similar to those for "Right" outcome model and have been omitted for clarity. Plotting symbols far from their corresponding shaded rectangles suggest that the SE estimation process may be flawed for that estimator in at least that particular scenario. (b) and (c) Empirical coverage rates for nominal 95% confidence intervals, with Wilson 95% confidence intervals shown for the empirical coverage rates. Grey band between .94 and .96 indicates our subjective "adequate performance zone" for a nominal 95% confidence interval. (c) shows results for only the unbiased estimators. PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability weighting.

the paired t-test provided slightly conservative estimates of the standard error under either propensity score model; this last result is consistent with the findings of Austin and Small (2014). For IPW, we see a similar pattern, with the standard error estimate for the IPW estimators that did not use further covariate adjustment being considerably more conservative than those for the unadjusted OOM estimator.

The figure shows two methods of standard error estimation for BOOM: the rows labelled "BOOM, BC" show Efron's bias-corrected estimates (Efron, 2014), and the rows labelled "BOOM, no BC" show the corresponding estimates without bias cor-

rection (Section 4.2.2). Consistent with the exposition in Efron (2014), the estimates without the bias correction are conservative in all cases. Under both propensity score models, the bias correction slightly over-corrects the standard error estimates for the covariate-adjusted BOOM, and slightly under-corrects the standard error estimates for the unadjusted BOOM, but does clearly improve the accuracy of each estimate.

4.4.5 Coverage of nominal 95% confidence intervals

Figures 4.4(b) and (c) show the empirical coverage of the nominal 95% confidence intervals for the estimators in the simulation study. In each panel, the grey band behind the plotting symbols indicates the region between 0.94 and 0.96, the zone in which we would consider the performance of a nominal 95% confidence interval to be "good enough." As in Figure 4.4(a), in each panel we present two rows for the BOOM estimator, corresponding to the confidence intervals constructed with and without the bias-corrected standard error estimates. In Figure 4.4(b) we see the low and extremely low coverage rates, due to bias, of the OLS estimators in the "No PS Model" scenarios with misspecified or no outcome model. Figure 4.4(c) presents the coverage results for the five scenarios that yielded unbiased estimators, those for which at least one of the two models (treatment or outcome) is correct, so that we can compare those more closely. As with mean squared error and variance, the whole-sample OLS estimator using the correct outcome model is our gold standard here, with empirical coverage falling almost exactly at 95%. Looking at the other established estimators, we see that when at least one model is specified correctly, the coverage rates for all the OOM estimators are consistent with a true coverage rate of 95%, and those for some, but not all, of the IPW estimators are consistent with the "good enough" zone. For the BOOM confidence intervals, in the "Right PS Model, No Outcome Model" scenario we see much better performance for the interval constructed using the bias-corrected standard error estimate than for the one constructed with the uncorrected estimate; the BOOM confidence interval using the bias-corrected standard error estimate appears consistent with a true coverage rate of 95% in this scenario. For the BOOM scenarios in which the correct or slightly misspecified outcome model is used and the estimator is unbiased, the empirical coverage rates are consistent with the "good enough" zone regardless of whether the bias-corrected standard error is used. The empirical coverage rates for the intervals constructed without the bias correction are slightly closer to 95%, but their coverage is conservative.

4.4.6 Results from other prevalence levels

As mentioned in Section 4.3.1.2, although we ran all simulations under three treatment prevalences, above we presented the results only for the simulations with average treatment prevalence of 10%. Results for the other two prevalences are in Appendix B (Section 4.9), and we summarize the key findings here. For mean squared error and variance, we see similar patterns across the three prevalence levels, but we note that the relative advantages of BOOM over OOM decreased slightly with increasing treatment prevalence (that is, with decreasing number of controls to choose from). In terms of bias, patterns are once again similar across prevalence levels, but under a slightly misspecified propensity score model the scaled bias of the BOOM, OOM, and IPW estimators increases very slightly with increasing treatment prevalence. In fact, at treatment prevalence of 20%, the IPW estimator using a misspecified propensity score model and no outcome model was no longer less biased than the OLS estimator with a misspecified outcome model. Here, then, we have a case where, as Kang and Schafer (2007) put it, "in at least some settings, two wrong models are not better than one." The OOM and BOOM estimators, however, had lower bias than the misspecified OLS estimator at all treatment prevalence levels, even with two misspecified models.

In comparing the accuracy of the standard error estimates across treatment prevalence levels, we note first that the standard deviations of all the treatment effect estimates decrease with increased treatment prevalence. While the bias correction for the BOOM standard error estimate seems to be clearly helpful at treatment prevalences of 5% and 10%, the situation is less clear at 20%, where the bias correction improved estimation for the BOOM estimators that did not use further covariate adjustment, but resulted in a less accurate standard error estimate for the BOOM estimators that incorporated outcome modeling. With the exception of the IPW estimators, the patterns of confidence interval performance were similar across the treatment prevalences. At each level of treatment prevalence, some, but not all, of the IPW confidence intervals were consistent with the "good enough" zone, with performance generally improving with increased treatment prevalence. At 20% treatment prevalence, two of the IPW intervals were consistent with a true coverage rate of 95%.

4.5 Case study

Here we demonstrate BOOM estimation using data analyzed by Connors et al. (1996) and publicly available at http://biostat.mc.vanderbilt.edu/DataSets. In con-

trast to the careful clinical analysis presented by Connors et al., here we use the data for purely illustrative purposes. We consider 5,734 hospital patients from the Study to Understand Prognoses and Preferences for Outcomes and Risks of Treatments (SUPPORT), 2,183 of whom received right heart catheterization (RHC) within the first 24 hours after study entry and 3,551 of whom did not, and we compare the two groups on average length of stay in the hospital, using the methods from the simulation study presented in Section 4.3 (OLS, BOOM, OOM, and IPW). For simplicity, rather than using the full set of covariates used by Connors et al., we use the following covariates in our propensity score and outcome models: type of medical insurance, primary disease category, secondary disease category, presence of neurological diagnosis at admission, do-not-resuscitate status, SUPPORT model estimate of the probability of surviving two months, Glasgow Coma Score, mean blood pressure, hematocrit, sodium, history of renal disease, and history of upper GI bleeding.

The use of a single dataset (rather than 5,000, as in the simulation study) allows us to look more closely at the BOOM process, particularly at the way in which individual subjects contribute to the BOOM estimate. We set $B = 10{,}000$ bootstrap resamples, a relatively low number for the purposes of BOOM standard error estimation in a dataset of this size, given the recommendations in Section 4.2.1.5. Patients were included in a bootstrap sample between 9,635 and 10,367 times, with the average being 10,000 (as expected). 639 treated and 404 control patients were matched to a subject from the opposite group every time they appeared in a bootstrap resample; these 1,043 patients received a BOOM weight of one, the highest possible BOOM weight (Section 4.2.5). The control patient with the lowest propensity score in the original sample was never included in a matched sample, despite appearing in a bootstrap sample 9,830 times. This patient was the only patient in the sample to receive a BOOM weight of zero. The mean BOOM weight was about 0.63, and the standard deviation of the BOOM weights was about 0.34.

In contrast to BOOM's use of almost every patient in this sample, our one-to-one matching algorithm matched 1,821 treated patients to 1,821 control patients, effectively giving those 3,642 patients a weight of one and the remaining patients a weight of zero; and under the IPW ATT weighting, each treated patient received a weight of one, and the weights for the control group ranged from approximately 0.014 to 9.2, with a mean of about 0.77 and standard deviation of about 0.55.

As in the full analysis by Connors et al., in our analysis, all methods suggested that patients given RHC had longer hospital length-of-stay than patients not given RHC. BOOM, OOM, IPW, and covariate adjustment on the whole sample (OLS)

gave results ranging from 1.9 (IPW) to to 2.9 (OLS) extra days in the hospital, estimates relatively close to each other in comparison to the unadjusted difference in means, which was 5.2 days. Length of 95% confidence intervals, all of which excluded zero, ranged from 2.9 days for the OLS estimate to 3.7 days for the IPW estimate with no covariate adjustment. For the BOOM estimates, we constructed the confidence intervals using the bias-corrected standard error estimates to compensate for the relatively small value of B we used. The confidence interval for the BOOM estimate with further covariate adjustment was just 5% longer than the confidence interval for the whole-sample OLS estimate (the most efficient estimate possible if the model is correct). The confidence intervals for the BOOM estimates with and without further covariate adjustment were 89–90% of the length of the confidence interval for the corresponding OOM estimate. The BOOM confidence intervals were also shorter than the IPW intervals: just 83–85% of the length of the IPW confidence intervals. The comparison between BOOM and IPW, however, may not be a fair one; in contrast to our tightly controlled simulation study where BOOM and IPW had exactly the same estimand, in this example dataset the two estimands are not necessarily the same.

4.6 Discussion

4.6.1 Advantages of the new method

The BOOM estimator shows promise as a tool for reducing bias in the estimation of continuous treatment effects. In our simulation study it retained the bias-reducing abilities of one-to-one matching, while showing great improvement over one-to-one matching in terms of MSE and variance, and it performed as well as or better than inverse probability weighting on all dimensions studied. In our case study it showed similarly promising performance.

In our simulations, with a good propensity score model, the BOOM estimator did almost as well in terms of MSE as if we had known the true outcome model and used every single data point. On first glance this might not seem like such an advantage; is an analyst any more likely to come up with a good propensity score model than with the true outcome model? We argue that the answer is yes. The flexible modeling techniques and detailed residual analyses essential for specifying and evaluating a good outcome model (Harrell, 2015) are unfortunately underused in common practice. The same modeling techniques should be utilized for the propensity score model and commonly are not. However, the key measure of how the propensity

score model performs in is the covariate balance it produces. As Austin (2011) argues, it is easier to assess covariate balance in the matched or weighted sample than it is to evaluate the specification and fit of the outcome model. Indeed, researchers typically present some exploration of covariate balance in detail as the first table in their paper.

Even with a slightly misspecified propensity score model, the BOOM estimator still did quite well in terms of MSE in our simulations; and with a slightly misspecified propensity score model and a missing or slightly misspecified outcome model, the BOOM estimator was less biased than using ordinary least squares estimation on the whole sample with a slightly misspecified outcome model. Because varying degrees of model misspecification are a common occurrence in statistical analysis, we believe that BOOM can play a key role in reducing bias while keeping MSE and variance low. However, we hope the robustness of the BOOM estimator is not taken as an excuse to be lax in the specification and checking of models.

In our case study, the BOOM process provided estimates that were very close to estimates from currently available and trusted methods, and also close to the estimates from the canonical analysis of the same data. While we cannot know the true value of the average treatment effect in this data set, we find it encouraging that the BOOM results for the SUPPORT data are close to results from established methods and analyses. As in the simulations, in the case study BOOM provided gains in efficiency over one-to-one matching and possibly over IPW. Although we expected BOOM to be an improvement over one-to-one matching, we were surprised that it performed as well as IPW in terms of efficiency in both the simulations and the case study, because weighting uses every subject in the sample, but there is no guarantee that each subject will contribute directly to the BOOM estimate.

As noted in Section 4.2, the BOOM algorithm is highly adaptable. Here we highlight two variations in particular that provide for situations in which the analyst has a great deal of uncertainty about the best distance measure and/or outcome model to use. Analysts wishing to avoid specifying either a propensity score model or an outcome model can use the BOOM estimation process with Mahalanobis distance matching or one of its variations, e.g. Reweighted Mahalanobis Distance (Greevy et al., 2012), and no covariate adjustment. Analysts wanting to use propensity score models and further covariate adjustment but also to account for uncertainty in the specification of those models can incorporate model selection into every iteration, thus ensuring as in Efron (2014) that the standard error estimates reflect this uncertainty.

4.6.2 Limitations and future directions

Because of the large number of computations involved, BOOM estimation may currently be impractical with larger datasets. Also, even though the BOOM estimator is more efficient than one-to-one matching (OOM), some researchers may find the descriptive "Table 1" for the weighted cohort associated with the BOOM estimator less intuitive than the straightforward Table 1 that follows from OOM. While the method does not currently explicitly handle missing data, the incorporation of imputation into the BOOM process seems straightforward, and we would like to extend the method to datasets with missing data. We would also like to explore the variations suggested in Section 4.2.4 and elsewhere in Section 4.2. Unfortunately, some of the variations, such as the use of Mahalanobis distance matching rather than propensity score matching, do not lend themselves well to simulation studies because they are so time-intensive. In addition to investigating variations of the BOOM process, we would also like to evaluate the method in simulations using a more complex and realistic data-generation mechanism, including mixed data types, correlated covariates, and slightly heterogeneous treatment effects, as well as under additional types of model misspecification. Finally, we would like to develop the method for non-collapsible outcomes, in particular binary and time-to-event outcomes, and for studies with multiple treatment groups.

4.7 Acknowledgements

The authors are grateful to Meira Epplein, Bryan Shepherd, and Matt Shotwell for their comments and suggestions on this manuscript.

This work was conducted in part using the resources of the Advanced Computing Center for Research and Education at Vanderbilt University, Nashville, TN.

4.8 Appendix A. R code from simulation study

4.8.1 args.R (file containing arguments)

```
# Values used by the main RunSims function and the functions it calls

# N: number of subjects per simulated dataset
N <- 1000

# beta.0.treat.vec: The intercepts to use in the treatment-selection model.
# This will determine the prevalence of treatment in the simulated dataset.
beta.0.treat.vec<- c(-3.95, -3.05, -2.07) # to get (.05, .1, .2) treated

# Values of coefficients for treatment and outcome generation
beta.low        <- 0.25
beta.med        <- 0.50
beta.high       <- 0.75
beta.v.high     <- 0.90

rightOutcomeFormula <- y ~
    treat +
    pol(x.4, 2) +
    x.5 * x.6 +
    pol(x.7, 2) +
    x.8 + x.9 + x.10

# for the "right" PS formula we are doing as Austin & Small did,
#   using the variables that affect outcome
rightPSFormula <- treat ~
    pol(x.4, 2) +
    x.5 * x.6 +
    pol(x.7, 2) +
    x.8 + x.9 + x.10

# for the "wrong" models we leave out squared terms and interactions
wrongOutcomeFormula <- y ~
    treat +
    x.4 + x.5 + x.6 + x.7 + x.8 + x.9 + x.10

wrongPSFormula <- treat ~
    x.4 + x.5 + x.6 + x.7 + x.8 + x.9 + x.10
```

4.8.2 Functions

```
MakeDat <- function(N, true.avg.TE.cont, Beta.0,
        Beta.low, Beta.med, Beta.high, Beta.v.high){
    # return a data.frame of the covariates, true treatment effects,
    #   treatment indicators, & outcomes

    x.1  <- rnorm(N, 0, 1)
    x.2  <- rnorm(N, 0, 1)
    x.3  <- rnorm(N, 0, 1)
    x.4  <- rnorm(N, 0, 1)
    x.5  <- rnorm(N, 0, 1)
    x.6  <- rnorm(N, 0, 1)
    x.7  <- rnorm(N, 0, 1)
    x.8  <- rnorm(N, 0, 1)
    x.9  <- rnorm(N, 0, 1)
    x.10 <- rnorm(N, 0, 1)

    X <- data.frame(x.1 , x.2 , x.3 , x.4 , x.5 , x.6 , x.7 , x.8 , x.9 , x.10)

    # Generate treatment status for each subject
    p.treat <- GetTxProbs(X, Beta.0, Beta.low, Beta.med, Beta.high, Beta.v.high)
    X$treat <- rbinom(N, 1, p.treat)

    X$TE.cont <- rep(true.avg.TE.cont, N)

    # The AddY func returns a data.frame with Y added
    AddY(X, Beta.low, Beta.med, Beta.high, Beta.v.high)
}
```

```r
GetTxProbs <- function(X, beta.0, beta.low, beta.med, beta.high, beta.v.high){
    # Return treatment probability for each subject,
    #   according to the underlying prevalence beta.0.
    # X needs to have at least columns x.1...x.7
    logit <- with(X,
        beta.0 +

        # these 3 are associated with tx assignment but not outcome
        beta.low  * x.1 +
        beta.med  * x.2 +
        beta.high * x.3 +

        0.5 * beta.low  * x.4 +
        0.3 * beta.low  * x.4^2 +

        0.5 * beta.med  * x.5 +
        0.5 * beta.high * x.6 +
        0.3 * beta.high  * x.5 * x.6 +

        0.5 * beta.v.high * x.7 +
        0.3 * beta.v.high * x.7^2
    )
    exp(logit) / (1 + exp(logit))
}
```

```r
AddY <- function(X, beta.low, beta.med, beta.high, beta.v.high){
    # return X with an additional column, y
    # X must have at least the columns x.4...x.10 and TE.cont (true treatment effect)

    lp.core <- with(X,
        0.5 * beta.low  * x.4 +
        0.3 * beta.low  * x.4^2 +

        0.5 * beta.med  * x.5 +
        0.5 * beta.high * x.6 +
        0.3 * beta.high  * x.5 * x.6 +

        0.5 * beta.v.high * x.7 +
        0.3 * beta.v.high * x.7^2 +

        # these three are associated with outcome but not tx assignment
        beta.low * x.8 +
        beta.med * x.9 +
        beta.high * x.10
    )

    # Generate continuous outcome for each subject
    y <- with(X,
        TE.cont * treat +
        lp.core +
        rnorm(nrow(X), 0, 3))

    data.frame(X, y)
}
```

```r
GetLMResults <- function(dat, form, true.TE.cont= NULL, justTxEst= FALSE, wts= NULL){
    # returns the TE estimate, SE estimate, and coverage.
    # wts, if used, is a string giving name of weights column in dat

    if(is.null(wts)){
        fit <- lm(form, data= dat)
    } else {
        svyobj <- svydesign(ids= ~0, weights= dat[[wts]], data= dat)
        fit <- svyglm(form, svyobj)
    }

    if(justTxEst){
        coef(fit)['treat']
    } else {
        c(  coef(fit)['treat'],
            sqrt(vcov(fit)['treat', 'treat']),
            # from confint.svyglm help: The default is a Wald-type confidence interval,
            #   adding and subtracting a multiple of the standard error.
            confint(fit)['treat', 1] <= true.TE.cont &&
                true.TE.cont <= confint(fit)['treat', 2])
    }
}
```

```
GetLogitPS <- function(dat, lrm.formula){
    fit <- tryCatch(
        lrm(lrm.formula, data= dat),
        error= function(e) return(NULL)
    )
    if(is.null(fit)) return(NULL)

    fit$linear.predictors
}
```

```r
GetPairs <- function(treatvec, logit.ps){
    # Return a matrix of pairs, with tx rownums in col 1 & ctrl rownums in col 2
    # Return NULL if PS estimation fails or no matches found

    # treatvec is a vector of treatment assignments (0's and 1's)
    # logit.ps is a vector of logit propensity scores

    # uses Austin & Small (2014) method 2: greedy NNM on logit of PS
    #    within calipers of width equal to 0.2 * sd(logit(PS))
    #    and using random ordering of treated subjects

    # data is already in random order bec. that's how it was generated.
    # Match() processes treated obsns in order they are presented, so it's
    # processing treated obsns in random order.

    if(is.null(logit.ps)) return(NULL)

    mm <- tryCatch(
        Match(
            Y       = NULL,
            Tr      = treatvec,
            X       = logit.ps,
            replace = FALSE,
            M       = 1, # the ratio
            ties    = FALSE, # randomly break ties
            caliper = 0.2,
            version = "fast"
        ),
        error= function(e) return(NULL)
    )
    if(is.null(mm) | length(mm$index.treated) == 0) return(NULL)

    ## return the pairs. Tx indices in col 1, ctrl in col 2
    cbind(mm$index.treated, mm$index.control)
}
```

```r
EfronEstimates <- function(count.matrix, est.TEs){
    # returns regular and bias-corrected estimates of SE
    #   for the estimator that is the mean of the est.TEs, using formulas from
    #   Efron 2014 JASA: Estimation and Accuracy After Model Selection

    # count.matrix is the matrix of bootstrap inclusion counts,
    #   with nrow= number of bootstraps and ncol= number of subjects in original sample,
    #   and no missing values
    # est.TEs is the vector of estimated treatment effects,
    #   with length = nrow(count.matrix)

    # In case there were boot samples where the matching failed:
    indicesToKeep <- rowSums(count.matrix) != 0
    count.matrix <- count.matrix[indicesToKeep, ]
    est.TEs <- est.TEs[indicesToKeep]

    # Number of people
    N <- ncol(count.matrix)
    # Number of bootstrap resamples (that actually worked)
    n.boot <- nrow(count.matrix)

    # Avg count for each person
    Y.star.dot.js <- colMeans(count.matrix)

    est.TE <- mean(est.TEs)
    # the centered means. Vector of length n.boot
    second.term.vec <- est.TEs - est.TE

    est.cov.efron <- est.cov.whe <- rep(NA, N)
    for(j in 1:N){
        # from Efron 2014:
        est.cov.efron[j] <-
            sum((count.matrix[, j] - Y.star.dot.js[j]) * second.term.vec) / n.boot

        # from Wager Hastie Efron (WHE) 2014:
        # (also used in Efron BC estimate)
        est.cov.whe[j] <-
            sum((count.matrix[, j] - 1) * second.term.vec) / n.boot
    }
    est.var.cont.efron <- sum(est.cov.efron^2)

    # for bias-corrected versions
```

```
    Z.matrix <- diff.matrix <- matrix(0, nrow= n.boot, ncol= N)
    #Z.matrix.alt <- diff.matrix.alt <- matrix(0, nrow= n.boot, ncol= N)

    for(i in 1:n.boot){
        for(j in 1:N){
            Z.matrix[i, j] <-
                (count.matrix[i, j] - 1) * second.term.vec[i]
            diff.matrix[i, j] <-
                Z.matrix[i, j] - est.cov.whe[j]

            # Does it make a difference if we use mean count
            #   rather than expected count? No, but leaving code here as reference
            #Z.matrix.alt[i, j] <-
            #    (count.matrix[i, j] - Y.star.dot.js[j]) * second.term.vec[i]
            #diff.matrix.alt[i, j] <-
            #    Z.matrix.alt[i, j] - est.cov.efron[j]
        }
    }
    est.var.cont.efron.bc <- est.var.cont.efron -
        (1 / n.boot^2) * sum(rowSums(diff.matrix^2))
    #est.var.cont.efron.bc.alt <- est.var.cont.efron -
    #    (1 / n.boot^2) * sum(rowSums(diff.matrix.alt^2))
    # From WHE p. 1629 eq (7). This also returns same result; leaving code as reference
    #est.var.cont.efron.bc.whe <- est.var.cont.efron -
    #    (N / n.boot^2) * sum(second.term.vec^2)

    c(est.var.cont.efron, est.var.cont.efron.bc#,
        #est.var.cont.efron.bc.alt, est.var.cont.efron.bc.whe
    )
}
```

```r
BOOMForSims <- function(dat, n.boot, ps.formula.list, lm.formula.correct= NULL,
    lm.formula.incorrect= NULL, catfname= NULL, seed= NULL, mcCores= 2,
    tx.indicator= "treat", outcome= "y"){
    # Returns a vector of various summary values from the BOOM procedure

    # dat: the original dataset
    # n.boot: number of bootstrap resamples to use
    # ps.formula.list: list of propensity score formulas to use w/ lrm
    # lm.formula.correct and .incorrect: optional outcome formulas
    # catfname: name of already-existing file to add messages to (optional)
    # seed is not used to set anything-- it's for use in the messages
    #    written to catfname; optional
    #    (the real seed is set in the function that calls this one)
    # mcCores: number of cores for mclapply()
    # tx.indicator: name of the treatment indicator variable in dat (1/0)
    # y: name of the continuous outcome variable in dat

    N <- nrow(dat) # tot number of subjects

    tx.dat <- dat[dat[[tx.indicator]] == 1, ]
    tx.ids <- 1:nrow(tx.dat)

    ctrl.dat <- dat[dat[[tx.indicator]] == 0, ]
    ctrl.ids <- (nrow(tx.dat) + 1) : N

    nforms <- length(ps.formula.list)
    bootStuff <- mclapply(1:n.boot, function(x) {
        # Modified from Austin & Small (2014) --- they did not condition on
        # observed tx & ctrl group sizes
        tx.sample.indices <-
            sample(1:nrow(tx.dat), size= nrow(tx.dat), replace= TRUE)
        tx.sample <- tx.dat[tx.sample.indices, ]

        ctrl.sample.indices <-
            sample(1:nrow(ctrl.dat), size= nrow(ctrl.dat), replace= TRUE)
        ctrl.sample <- ctrl.dat[ctrl.sample.indices, ]

        boot.sample <- rbind(tx.sample, ctrl.sample, make.row.names= FALSE)

        est.mean.tx.tmp <- est.mean.ctrl.tmp <-
            est.TE.lm.tmp <- est.TE.lm.incorrect.tmp <- rep(NA, nforms)
```

```
    # well, count.vector used to be a vector.
    #    Now it's multiple row-vectors, one per PS formula. Keeping the name
    count.vector.tmp <- matrix(0, nrow= nforms, ncol= N)
    num.errs.tmp <- rep(0, nforms)

    for (formNum in 1:length(ps.formula.list)) {
        logitPS <- GetLogitPS(boot.sample, ps.formula.list[[formNum]])
        pairIndices <- GetPairs(boot.sample[[tx.indicator]], logitPS)
        # tx indices are in 1st col, ctrl in 2nd col
        # We are handling PS estimation/matching errors by skipping that resample.
        #    Maybe not the best way, but it rarely happens.
        if(!is.null(pairIndices)){
            est.mean.tx.tmp[formNum] <-
                mean(boot.sample[pairIndices[, 1], outcome])
            est.mean.ctrl.tmp[formNum] <-
                mean(boot.sample[pairIndices[, 2], outcome])

            if (!is.null(lm.formula.correct)) {
                est.TE.lm.tmp[formNum] <- GetLMResults(
                    boot.sample[c(pairIndices[, 1], pairIndices[, 2]), ],
                    lm.formula.correct, justTxEst= TRUE
                )
            }

            if (!is.null(lm.formula.incorrect)) {
                est.TE.lm.incorrect.tmp[formNum] <- GetLMResults(
                    boot.sample[c(pairIndices[, 1], pairIndices[, 2]), ],
                    lm.formula.incorrect, justTxEst= TRUE
                )
            }

            # fill in Efron counts
            both.tbl <- c(
                table(tx.sample.indices),
                table(ctrl.ids[ctrl.sample.indices])
            )
            count.vector.tmp[formNum, as.numeric(names(both.tbl))] <-
                both.tbl
        } else { # we did not get a match in this resample
            if (!is.null(catfname) & !is.null(seed)) {
            cat("Hit problem in boot.\n", seed, "\n",
                file = catfname, append= TRUE)
            }
```

```
            num.errs.tmp[formNum] <- num.errs.tmp[formNum] + 1
        } # end processing for resamples with errors in PS estimation or matching
    } # end this PS formula
    # return from mclapply:
    list(
        # the first four are vectors w/ length = number of PS models
        est.mean.tx.tmp, #1
        est.mean.ctrl.tmp, #2
        est.TE.lm.tmp, #3
        est.TE.lm.incorrect.tmp, #4

        # As noted above, despite its name, this one is a matrix
        count.vector.tmp, #5

        # vector w/ length = number of PS models
        num.errs.tmp #6
        )
},
    mc.cores           = mcCores,
    mc.preschedule     = TRUE,
    mc.set.seed        = TRUE,
    mc.allow.recursive = FALSE
) # end bootstrap resampling (mc)lapply

# one row per PS formula
est.means.tx   <- sapply(bootStuff, function(x) x[[1]])
est.means.ctrl <- sapply(bootStuff, function(x) x[[2]])
est.TEs <- est.means.tx - est.means.ctrl
est.TEs.lm <- sapply(bootStuff, function(x) x[[3]])
est.TEs.lm.incorrect <- sapply(bootStuff, function(x) x[[4]])

# there are fancier ways to do this, but I get confused
# Array: row = resample; col = subject; table = formula
count.array <- array(0, c(n.boot, N, nforms))
for (formNum in 1:length(ps.formula.list)) {
    count.array[, , formNum] <-
        do.call(rbind, lapply(bootStuff, function(x) x[[5]][formNum, ]))
}
count.array.tx <- count.array[, tx.ids, ]
count.array.ctrl <- count.array[, ctrl.ids, ]

# one row per PS formula
num.errs <- sapply(bootStuff, function(x) x[[6]])
```

```
# Summary stats (bagged statistics)
est.TE <- rowMeans(est.TEs, na.rm= TRUE)
est.TE.lm <- rowMeans(est.TEs.lm, na.rm= TRUE)
est.TE.lm.incorrect <- rowMeans(est.TEs.lm.incorrect, na.rm= TRUE)

tot.errs <- rowSums(num.errs, na.rm= TRUE)

# Efron calculations: Regular & BC
#   there are fancier ways to do this, but I get confused
NUM.THINGS.RETURNED.BY.EE <- 2
efron <- efron2.tx <- efron2.ctrl <- efron.lm <- efron.lm.incorrect <-
    matrix(NA, ncol= NUM.THINGS.RETURNED.BY.EE, nrow= nforms)
for (i in 1:nforms) {
    efron2.tx[i, ] <-
        EfronEstimates(count.array.tx[, , i], est.means.tx[i, ])
    efron2.ctrl[i, ] <-
        EfronEstimates(count.array.ctrl[, , i], est.means.ctrl[i, ])
    efron.lm[i, ] <-
        EfronEstimates(count.array[, , i], est.TEs.lm[i, ])
    efron.lm.incorrect[i, ] <-
        EfronEstimates(count.array[, , i], est.TEs.lm.incorrect[i, ])
}


est.SE <- apply(est.TEs, 1, sd, na.rm= TRUE)
est.SE.lm <- apply(est.TEs.lm, 1, sd, na.rm= TRUE)
est.SE.lm.incorrect <- apply(est.TEs.lm.incorrect, 1, sd, na.rm= TRUE)

cbind(
    est.TE,
    est.SE,  # not recommended for use; just for comparison
    sqrt(efron2.tx[, 1] + efron2.ctrl[, 1]),
    sqrt(efron2.tx[, 2] + efron2.ctrl[, 2]),

    est.TE.lm,
    est.SE.lm,  # not recommended for use; just for comparison
    sqrt(efron.lm[, 1]),
    sqrt(efron.lm[, 2]),

    est.TE.lm.incorrect,
    est.SE.lm.incorrect,  # not recommended for use; just for comparison
    sqrt(efron.lm.incorrect[, 1]),
```

```
        sqrt(efron.lm.incorrect[, 2]),

        tot.errs
    )
}
```

```r
RunSims <- function(
    codedir,
    outputdir,
    n.iter              = 10,
    n.boot              = 10,
    init.seed           = 57693,
    mcCores             = 2
    ) {
    ################################################################
    # Run the sims and save results.
    # Source and call this from another file.

    # codedir: the directory where this file and the other .R files are kept
    # outputdir: the directory where the .csv output and .txt record should be saved
    # n.iter: number of simulated datasets to make
    # n.boot: number of bootstrap resamples used by BOOM
    # init.seed: starting seed
    # mcCores: number of cores to be used by mclapply() in the BOOM process

    ################################################################

    source(file.path(codedir, "args.R"))

    ################################################################
    # Save results to outputdir with following name ('.csv' will be added later):
    fname <- sprintf("seed%i_N%i_it%i_b%i", init.seed, N, n.iter, n.boot)
    # can use this file to track progress, etc.
    catfname <- file.path(outputdir, paste0(fname, "_record.txt"))
    ################################################################

    ################################################################

    # Things we'll return (one set for every PS model-tx prevalence combination
    #    within each iteration)
    thingsToReturn <- c(
        # This set does not depend on either model specification
        'iteration',
        'seed',
        'beta.0.treat',
        'prop.treated.orig',
        'n.treated.orig',
```

```
        'est.TE.ttest.all',
        'est.SE.ttest.all',
        'covered.ttest.all',

        # This set depends on outcome model spec., but not PS model spec
        'est.TE.lm.correct.all',
        'est.SE.lm.correct.all',
        'covered.lm.correct.all',

        'est.TE.lm.incorrect.all',
        'est.SE.lm.incorrect.all',
        'covered.lm.incorrect.all',

        # the following depend on the PS model specification
        'PSModel',
        'est.TE.nocov.iptw.att',
        'est.SE.nocov.iptw.att',
        'covered.nocov.iptw.att',

        # This set depends on both models
        'est.TE.lm.correct.iptw.att',
        'est.SE.lm.correct.iptw.att',
        'covered.lm.correct.iptw.att',

        'est.TE.lm.incorrect.iptw.att',
        'est.SE.lm.incorrect.iptw.att',
        'covered.lm.incorrect.iptw.att',

        # In addition to depending on PS model, these depend on successful matching
        'n.treated.final',
        'est.TE',
        'est.SE.ttest.twosample',
        'covered.ttest.twosample',

        'est.SE.ttest.paired',
        'covered.ttest.paired',

        # And these depend on both models plus on successful matching
        'est.TE.lm.correct',
        'est.SE.lm.correct',
        'covered.lm.correct',

        'est.TE.lm.incorrect',
```

```
        'est.SE.lm.incorrect',
        'covered.lm.incorrect',

        # And the BOOM results (cx in names is for 'complex bootstrap'):
        'est.TE.cx',
        'est.SE.cx',
        'est.SE.cx.efron2',
        'est.SE.cx.efron2.bc',

        'est.TE.cx.lm.correct',
        'est.SE.cx.lm.correct',
        'est.SE.cx.lm.correct.efron',
        'est.SE.cx.lm.correct.efron.bc',

        'est.TE.cx.lm.incorrect',
        'est.SE.cx.lm.incorrect',
        'est.SE.cx.lm.incorrect.efron',
        'est.SE.cx.lm.incorrect.efron.bc',

        'num.errs.boot1'
    )

    print(date())

    # formulas are in args.R
    psFormulaList <- list(rightPSFormula, wrongPSFormula)
    names(psFormulaList) <- c('right', 'wrong')

    # Set up the average treatment effects. For now, using constant TE.
    true.avg.TE <- 1

    for(iter in 1:n.iter) {
        # dataset creation code modified from Austin & Small 2014 Stat in Med

        # Choose seed for random number generation within each iteration
        #   so that the datasets are reproducible
        #   regardless of matching method, bootstrap, etc.
        seed <- init.seed + iter - 1
        # we will set seed later, for each dset in the iteration

        cat("Starting seed: ", seed, " \n",
            file= catfname,
            append= ifelse(iter == 1, FALSE, TRUE)
```

```
    )

    resThisIter <- matrix(NA,
        nrow= length(beta.0.treat.vec) * length(psFormulaList),
        ncol= length(thingsToReturn))
    colnames(resThisIter) <- thingsToReturn

    # fill in the results that apply to all tx prevalences for this iteration
    resThisIter[, 'iteration'] <- rep(iter, nrow(resThisIter))
    resThisIter[, 'seed']      <- rep(seed, nrow(resThisIter))
    resThisIter[, 'PSModel']   <-
        rep(names(psFormulaList), each= length(beta.0.treat.vec))

    # beta.0.treat.vec is in args.R; one value per tx prevalence level
    for(betaNum in seq_along(beta.0.treat.vec)){
        beta.rownums <- c(betaNum, betaNum + length(beta.0.treat.vec))

        beta.0.treat <- beta.0.treat.vec[betaNum]
        resThisIter[beta.rownums, 'beta.0.treat'] <- beta.0.treat

        # Create the dataset
        set.seed(seed, kind= "L'Ecuyer-CMRG") # for mclapply()
        mc.reset.stream() # to keep results reproducible

        # beta.low, beta.med, etc. are in args.R
        dat <- MakeDat(N,
            true.avg.TE = true.avg.TE,
            Beta.0           = beta.0.treat,
            Beta.low         = beta.low,
            Beta.med         = beta.med,
            Beta.high        = beta.high,
            Beta.v.high      = beta.v.high)

        # proportion treated
        resThisIter[beta.rownums, 'prop.treated.orig'] <- mean(dat$treat)
        # number treated
        resThisIter[beta.rownums, 'n.treated.orig'] <- sum(dat$treat)

        ##########################################################
        # Whole-sample tests etc. that do not depend on estimated PS

        # t-test for whole sample
        y.tx.all   <- dat[dat$treat == 1, "y"]
```

```
    y.ctrl.all <- dat[dat$treat == 0, "y"]
    test4 <- t.test(y.tx.all, y.ctrl.all, paired= FALSE, var.equal= FALSE)
    est.TE.all <- test4$estimate[1] - test4$estimate[2]
    resThisIter[beta.rownums, 'est.TE.ttest.all'] <- est.TE.all
    resThisIter[beta.rownums, 'est.SE.ttest.all'] <-
        est.TE.all / test4$statistic
    resThisIter[beta.rownums, 'covered.ttest.all'] <-
        as.numeric(
            test4$conf.int[1] <= true.avg.TE &&
            test4$conf.int[2] >= true.avg.TE
        )

    # linear regression, correct model on FULL COHORT
    #   Note we are estimating ATE here, not ATT
    resThisIter[beta.rownums, c(
        'est.TE.lm.correct.all',
        'est.SE.lm.correct.all',
        'covered.lm.correct.all')] <-
        rep(
            GetLMResults(dat, rightOutcomeFormula, true.TE= true.avg.TE),
            each= length(psFormulaList)
        )

    # linear regression, incorrect model on FULL COHORT
    #   Note we are estimating ATE here, not ATT
    resThisIter[beta.rownums, c(
        'est.TE.lm.incorrect.all',
        'est.SE.lm.incorrect.all',
        'covered.lm.incorrect.all')] <-
        rep(
            GetLMResults(dat, wrongOutcomeFormula, true.TE= true.avg.TE),
            each= length(psFormulaList)
        )



    ############################################################
    # Everything else requires a PS formula
    for (formNum in 1:length(psFormulaList)) {

        logitPS <- GetLogitPS(dat, psFormulaList[[formNum]])

        ############################################################
        # Whole-sample tests etc. that depend on estimated PS,
```

```
#   but not on matching

if(!is.null(logitPS)){
    # IPTW for ATT
    #   {Austin:2013fu}: Using weights equal to Z + [(1-Z)e/(1-e)]
    #   allows one to estimate the average treatment effect in the
    #   treated (ATT)
    PS <- exp(logitPS) / (1 + exp(logitPS))
    dat <- within(dat, {
        iptw.att <- treat + (1 - treat) * PS / (1 - PS)
        #iptw.ate <- treat / PS + (1 - treat) / (1 - PS)
    })

    resThisIter[beta.rownums[formNum], c(
        'est.TE.nocov.iptw.att',
        'est.SE.nocov.iptw.att',
        'covered.nocov.iptw.att')] <-
        GetLMResults(dat,
            y ~ treat,
            true.TE= true.avg.TE,
            wts= "iptw.att")

    resThisIter[beta.rownums[formNum], c(
        'est.TE.lm.correct.iptw.att',
        'est.SE.lm.correct.iptw.att',
        'covered.lm.correct.iptw.att')] <-
        GetLMResults(dat,
            rightOutcomeFormula,
            true.TE= true.avg.TE,
            wts= "iptw.att")

    resThisIter[beta.rownums[formNum], c(
        'est.TE.lm.incorrect.iptw.att',
        'est.SE.lm.incorrect.iptw.att',
        'covered.lm.incorrect.iptw.att')] <-
        GetLMResults(dat,
            wrongOutcomeFormula,
            true.TE= true.avg.TE,
            wts= "iptw.att")
}

#########################################################
# Tests, etc. that use a single matched sample from original cohort
```

```
        # tx indices are in 1st col, ctrl in 2nd col
        pairIndices <- GetPairs(dat$treat, logitPS)

        if(!is.null(pairIndices) & nrow(pairIndices) >= 2){
            y.tx   <- dat[pairIndices[, 1], "y"]
            y.ctrl <- dat[pairIndices[, 2], "y"]
            resThisIter[beta.rownums[formNum], 'n.treated.final'] <-
                length(y.tx)

            est.TE <- mean(y.tx - y.ctrl)
            resThisIter[beta.rownums[formNum], 'est.TE'] <- est.TE

            # Get est. SE & coverage from 2-sample t-test.
            #   I think Austin & Small 2014 do not say which variance assumption
            #   they use.  We are picking unequal-variance t-test.
            test1 <- t.test(y.tx, y.ctrl, paired= FALSE, var.equal= FALSE)
            resThisIter[beta.rownums[formNum], 'est.SE.ttest.twosample'] <-
                est.TE / test1$statistic
            resThisIter[beta.rownums[formNum], 'covered.ttest.twosample'] <-
                as.numeric(
                test1$conf.int[1] <= true.avg.TE &&
                test1$conf.int[2] >= true.avg.TE)

            # get est. SE and coverage from paired t-test
            test3 <- t.test(y.tx, y.ctrl, paired= TRUE)
            resThisIter[beta.rownums[formNum], 'est.SE.ttest.paired'] <-
                est.TE / test3$statistic
            resThisIter[beta.rownums[formNum], 'covered.ttest.paired'] <-
                as.numeric(
                test3$conf.int[1] <= true.avg.TE &&
                test3$conf.int[2] >= true.avg.TE)

            # Linear regression, correct model on paired data.
            #   We need the whole matched dataset here,
            #   not just the matched outcomes.
            dat.matched <- dat[c(pairIndices[, 1], pairIndices[, 2]), ]
            resThisIter[beta.rownums[formNum], c(
                'est.TE.lm.correct',
                'est.SE.lm.correct',
                'covered.lm.correct')] <-
                GetLMResults(dat.matched,
                    rightOutcomeFormula,
```

```
                    true.TE= true.avg.TE)

            # linear regression, incorrect model on paired data
            resThisIter[beta.rownums[formNum], c(
                'est.TE.lm.incorrect',
                'est.SE.lm.incorrect',
                'covered.lm.incorrect')] <-
                GetLMResults(dat.matched,
                    wrongOutcomeFormula,
                    true.TE= true.avg.TE)

        } else{
            cat("<2 matches\n", seed, "\n", file = catfname, append= TRUE)
        }
    }
    ##################################################################

    # BOOM (complex ("cx") bootstrap)
    resThisIter[beta.rownums, c(
        'est.TE.cx',
        'est.SE.cx',
        'est.SE.cx.efron2',
        'est.SE.cx.efron2.bc',

        'est.TE.cx.lm.correct',
        'est.SE.cx.lm.correct',
        'est.SE.cx.lm.correct.efron',
        'est.SE.cx.lm.correct.efron.bc',

        'est.TE.cx.lm.incorrect',
        'est.SE.cx.lm.incorrect',
        'est.SE.cx.lm.incorrect.efron',
        'est.SE.cx.lm.incorrect.efron.bc',

        'num.errs.boot1')] <-
        BOOMForSims(
            dat,
            n.boot,
            ps.formula.list= psFormulaList,
            lm.formula.correct= rightOutcomeFormula,
            lm.formula.incorrect= wrongOutcomeFormula,
            catfname= catfname, seed= seed,
            mcCores= mcCores)
```

```
        } # end this treatment prevalence level (betaNum)
        cat("Ending seed: ", seed, "\n", file = catfname, append= TRUE)

        write.table(resThisIter,
            file      = file.path(outputdir, paste0(fname, ".csv")),
            sep       = ",",
            row.names= FALSE,
            append    = ifelse(iter == 1, FALSE, TRUE),
            col.names= ifelse(iter == 1, TRUE, FALSE)
        )

    } # end this iteration (seed)
    print(date())
    print(sessionInfo())
    return(NULL)
}
```

4.9 Appendix B. Full results from simulation study

4.9.1 Figures for additional prevalence levels: Mean squared error, bias, and variance

Note that the figures for the scenarios with treatment prevalence of 10% appear in the main manuscript.

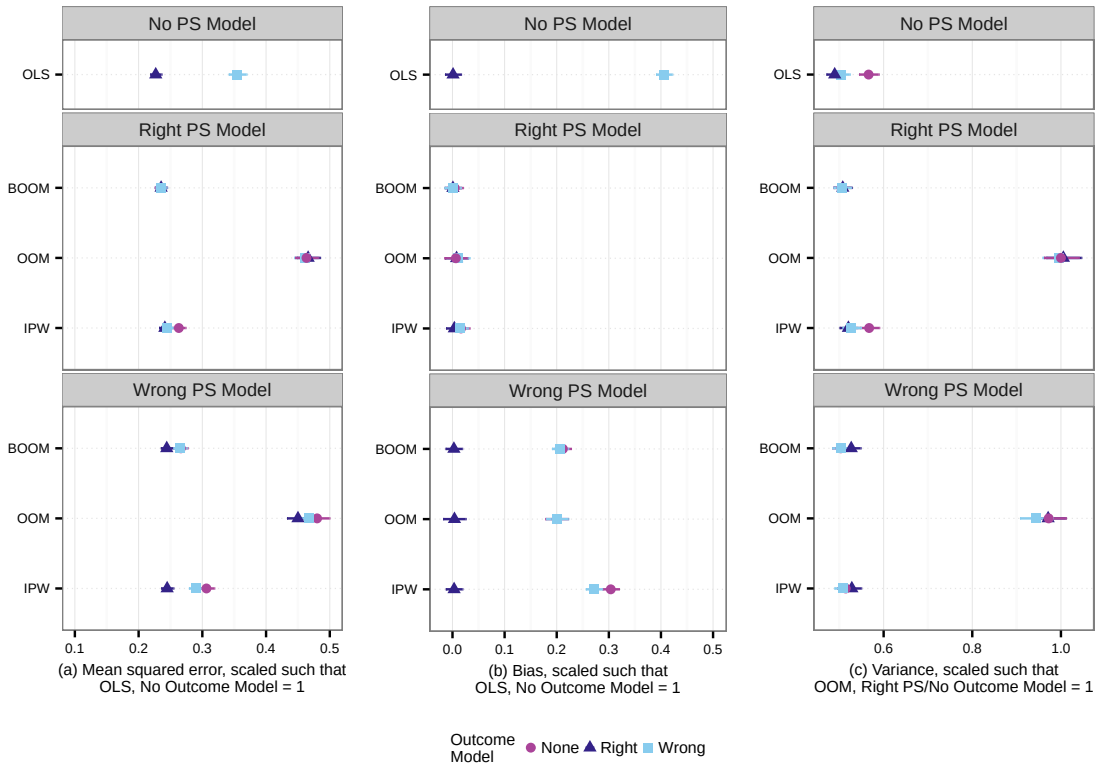

Figure 4.5: 5% treated. (a) Mean squared error, (b) bias, and (c) variance from simulation study. 95% confidence intervals are shown in all three panels. PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

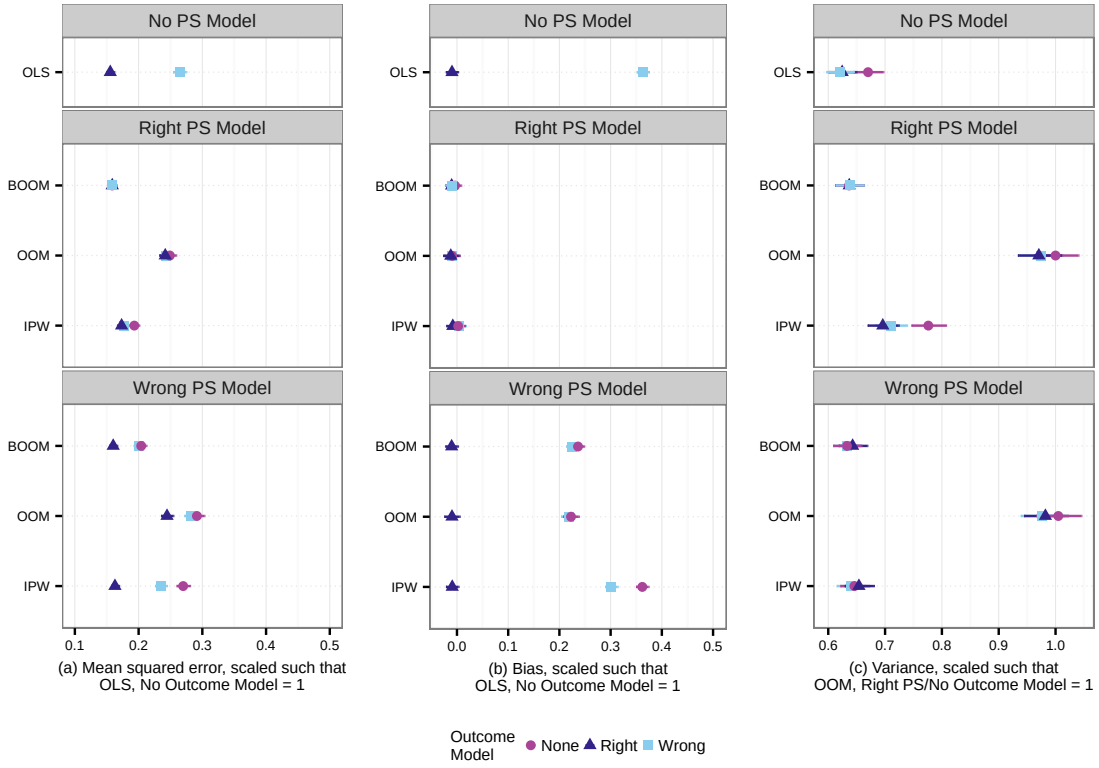

Figure 4.6: 20% treated. (a) Mean squared error, (b) bias, and (c) variance from simulation study. 95% confidence intervals are shown in all three panels. PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

4.9.2 Figures for additional prevalence levels: Accuracy of standard error estimates; coverage of nominal 95% confidence intervals

Note that the figures for the scenarios with treatment prevalence of 10% appear in the main manuscript.

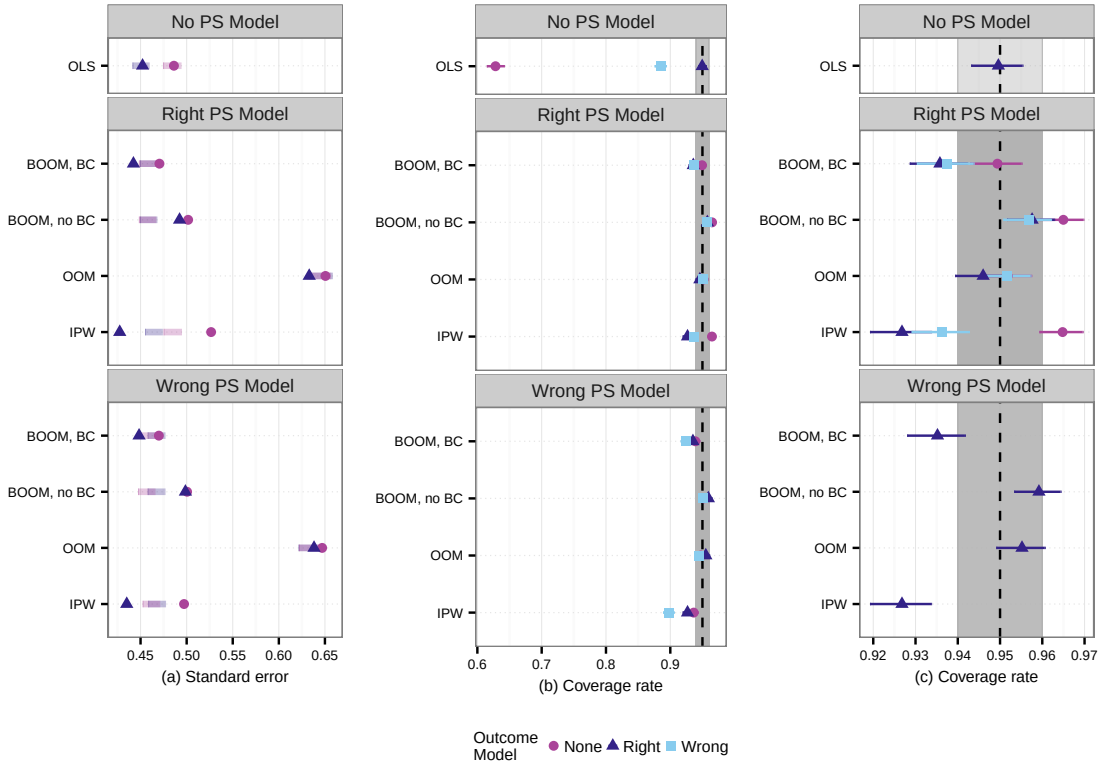

Figure 4.7: 5% treated. (a) Mean standard error (SE) estimates from simulation study; 95% confidence intervals for these estimates are narrower than the plotting symbols. 95% confidence interval for true SE in each case (actual standard deviation of the treatment effect estimates) is shown as a shaded rectangle behind the plotting symbols; confidence interval for true SE is due to Monte Carlo uncertainty. Results for "Wrong" outcome model are very similar to those for "Right" outcome model and have been omitted for clarity. Plotting symbols far from their corresponding shaded rectangles suggest that the SE estimation process may be flawed for that estimator in at least that particular scenario. (b) and (c) Empirical coverage rates for nominal 95% confidence intervals, with Wilson 95% confidence intervals shown for the empirical coverage rates. Grey band between .94 and .96 indicates our subjective "adequate performance zone" for a nominal 95% confidence interval. (c) shows results for only the unbiased estimators. PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability weighting.

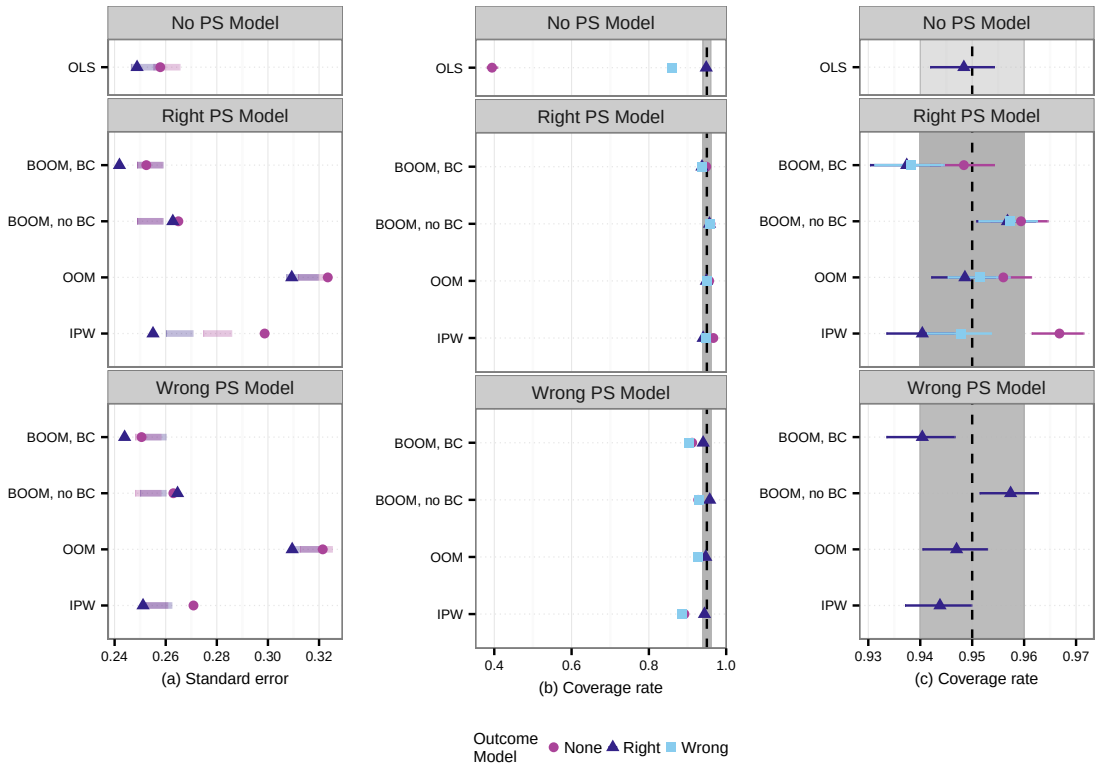

Figure 4.8: 20% treated. (a) Mean standard error (SE) estimates from simulation study; 95% confidence intervals for these estimates are narrower than the plotting symbols. 95% confidence interval for true SE in each case (actual standard deviation of the treatment effect estimates) is shown as a shaded rectangle behind the plotting symbols; confidence interval for true SE is due to Monte Carlo uncertainty. Results for "Wrong" outcome model are very similar to those for "Right" outcome model and have been omitted for clarity. Plotting symbols far from their corresponding shaded rectangles suggest that the SE estimation process may be flawed for that estimator in at least that particular scenario. (b) and (c) Empirical coverage rates for nominal 95% confidence intervals, with Wilson 95% confidence intervals shown for the empirical coverage rates. Grey band between .94 and .96 indicates our subjective "adequate performance zone" for a nominal 95% confidence interval. (c) shows results for only the unbiased estimators. PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability weighting.

4.9.3 Tables for all prevalence levels: Unscaled MSE

Table 4.2: 5% treated: Unscaled MSE, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.89 (0.86, 0.93)	0.20 (0.19, 0.21)	0.32 (0.30, 0.33)
Right PS Model				
	BOOM	0.21 (0.20, 0.22)	0.21 (0.20, 0.22)	0.21 (0.20, 0.22)
	OOM	0.41 (0.40, 0.43)	0.42 (0.40, 0.43)	0.41 (0.40, 0.43)
	IPW	0.23 (0.23, 0.24)	0.22 (0.21, 0.22)	0.22 (0.21, 0.23)
Wrong PS Model				
	BOOM	0.24 (0.23, 0.25)	0.22 (0.21, 0.23)	0.24 (0.23, 0.25)
	OOM	0.43 (0.41, 0.45)	0.40 (0.39, 0.42)	0.42 (0.40, 0.43)
	IPW	0.27 (0.26, 0.28)	0.22 (0.21, 0.23)	0.26 (0.25, 0.27)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.3: 10% treated: Unscaled MSE, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.60 (0.58, 0.62)	0.11 (0.10, 0.11)	0.18 (0.17, 0.18)
Right PS Model				
	BOOM	0.11 (0.10, 0.11)	0.11 (0.10, 0.11)	0.11 (0.10, 0.11)
	OOM	0.19 (0.19, 0.20)	0.19 (0.18, 0.20)	0.19 (0.18, 0.20)
	IPW	0.13 (0.12, 0.13)	0.11 (0.11, 0.12)	0.11 (0.11, 0.12)
Wrong PS Model				
	BOOM	0.13 (0.13, 0.14)	0.11 (0.11, 0.12)	0.13 (0.12, 0.13)
	OOM	0.21 (0.20, 0.22)	0.19 (0.18, 0.20)	0.21 (0.20, 0.22)
	IPW	0.16 (0.16, 0.17)	0.11 (0.11, 0.12)	0.15 (0.14, 0.15)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.4: 20% treated: Unscaled MSE, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.41 (0.39, 0.42)	0.06 (0.06, 0.07)	0.11 (0.10, 0.11)
Right PS Model				
	BOOM	0.06 (0.06, 0.07)	0.06 (0.06, 0.07)	0.06 (0.06, 0.07)
	OOM	0.10 (0.10, 0.11)	0.10 (0.09, 0.10)	0.10 (0.09, 0.10)
	IPW	0.08 (0.08, 0.08)	0.07 (0.07, 0.07)	0.07 (0.07, 0.07)
Wrong PS Model				
	BOOM	0.08 (0.08, 0.09)	0.06 (0.06, 0.07)	0.08 (0.08, 0.08)
	OOM	0.12 (0.11, 0.12)	0.10 (0.10, 0.10)	0.11 (0.11, 0.12)
	IPW	0.11 (0.11, 0.11)	0.07 (0.06, 0.07)	0.10 (0.09, 0.10)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

4.9.4 Tables for all prevalence levels: Unscaled bias

Table 4.5: 5% treated: Unscaled bias, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.81 (0.80, 0.82)	0.00 (-0.01, 0.01)	0.33 (0.32, 0.34)
Right PS Model				
	BOOM	0.00 (-0.01, 0.02)	0.00 (-0.01, 0.01)	0.00 (-0.01, 0.01)
	OOM	0.01 (-0.01, 0.02)	0.01 (-0.01, 0.02)	0.01 (-0.01, 0.03)
	IPW	0.01 (-0.00, 0.03)	0.00 (-0.01, 0.02)	0.01 (-0.00, 0.02)
Wrong PS Model				
	BOOM	0.17 (0.16, 0.18)	0.00 (-0.01, 0.02)	0.17 (0.15, 0.18)
	OOM	0.16 (0.14, 0.18)	0.00 (-0.01, 0.02)	0.16 (0.15, 0.18)
	IPW	0.25 (0.23, 0.26)	0.00 (-0.01, 0.02)	0.22 (0.21, 0.23)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.6: 10% treated: Unscaled bias, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.69 (0.68, 0.70)	-0.00 (-0.01, 0.01)	0.27 (0.26, 0.28)
Right PS Model				
	BOOM	0.00 (-0.01, 0.01)	-0.00 (-0.01, 0.01)	-0.00 (-0.01, 0.01)
	OOM	-0.00 (-0.01, 0.01)	-0.00 (-0.01, 0.01)	-0.00 (-0.01, 0.01)
	IPW	0.00 (-0.01, 0.01)	-0.00 (-0.01, 0.01)	0.01 (-0.00, 0.01)
Wrong PS Model				
	BOOM	0.16 (0.15, 0.16)	-0.00 (-0.01, 0.01)	0.15 (0.14, 0.16)
	OOM	0.15 (0.13, 0.16)	-0.00 (-0.01, 0.01)	0.15 (0.13, 0.16)
	IPW	0.23 (0.22, 0.24)	-0.00 (-0.01, 0.01)	0.20 (0.19, 0.21)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.7: 20% treated: Unscaled bias, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.58 (0.57, 0.59)	-0.01 (-0.01, 0.00)	0.21 (0.20, 0.22)
Right PS Model				
	BOOM	-0.00 (-0.01, 0.00)	-0.01 (-0.01, 0.00)	-0.01 (-0.01, 0.00)
	OOM	-0.01 (-0.01, 0.00)	-0.01 (-0.02, 0.00)	-0.01 (-0.01, 0.00)
	IPW	0.00 (-0.01, 0.01)	-0.00 (-0.01, 0.00)	0.00 (-0.00, 0.01)
Wrong PS Model				
	BOOM	0.14 (0.13, 0.14)	-0.01 (-0.01, 0.00)	0.13 (0.12, 0.14)
	OOM	0.13 (0.12, 0.14)	-0.01 (-0.01, 0.00)	0.13 (0.12, 0.14)
	IPW	0.21 (0.20, 0.22)	-0.01 (-0.01, 0.00)	0.18 (0.17, 0.18)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

4.9.5 Tables for all prevalence levels: Unscaled variance

Table 4.8: 5% treated: Unscaled variance, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.23 (0.23, 0.24)	0.20 (0.19, 0.21)	0.21 (0.20, 0.22)
Right PS Model				
	BOOM	0.21 (0.20, 0.22)	0.21 (0.20, 0.22)	0.21 (0.20, 0.22)
	OOM	0.41 (0.40, 0.43)	0.42 (0.40, 0.43)	0.41 (0.40, 0.43)
	IPW	0.23 (0.23, 0.24)	0.22 (0.21, 0.22)	0.22 (0.21, 0.23)
Wrong PS Model				
	BOOM	0.21 (0.20, 0.22)	0.22 (0.21, 0.23)	0.21 (0.20, 0.22)
	OOM	0.40 (0.39, 0.42)	0.40 (0.39, 0.42)	0.39 (0.38, 0.41)
	IPW	0.21 (0.20, 0.22)	0.22 (0.21, 0.23)	0.21 (0.20, 0.22)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.9: 10% treated: Unscaled variance, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.12 (0.11, 0.12)	0.11 (0.10, 0.11)	0.11 (0.10, 0.11)
Right PS Model				
	BOOM	0.11 (0.10, 0.11)	0.11 (0.10, 0.11)	0.11 (0.10, 0.11)
	OOM	0.19 (0.19, 0.20)	0.19 (0.18, 0.20)	0.19 (0.18, 0.20)
	IPW	0.13 (0.12, 0.13)	0.11 (0.11, 0.12)	0.11 (0.11, 0.12)
Wrong PS Model				
	BOOM	0.11 (0.10, 0.11)	0.11 (0.11, 0.12)	0.11 (0.10, 0.11)
	OOM	0.19 (0.18, 0.20)	0.19 (0.18, 0.20)	0.19 (0.18, 0.19)
	IPW	0.11 (0.10, 0.11)	0.11 (0.11, 0.12)	0.11 (0.10, 0.11)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.10: 20% treated: Unscaled variance, with 95% confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.07 (0.07, 0.07)	0.06 (0.06, 0.07)	0.06 (0.06, 0.07)
Right PS Model				
	BOOM	0.06 (0.06, 0.07)	0.06 (0.06, 0.07)	0.06 (0.06, 0.07)
	OOM	0.10 (0.10, 0.11)	0.10 (0.09, 0.10)	0.10 (0.09, 0.10)
	IPW	0.08 (0.08, 0.08)	0.07 (0.07, 0.07)	0.07 (0.07, 0.07)
Wrong PS Model				
	BOOM	0.06 (0.06, 0.07)	0.06 (0.06, 0.07)	0.06 (0.06, 0.07)
	OOM	0.10 (0.10, 0.11)	0.10 (0.10, 0.10)	0.10 (0.09, 0.10)
	IPW	0.07 (0.06, 0.07)	0.07 (0.06, 0.07)	0.06 (0.06, 0.07)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

4.9.6 Tables for all prevalence levels: Accuracy of standard error estimates

Table 4.11: 5% treated: Accuracy of standard error estimates. For each estimator, the first row gives the mean standard error estimate and the second row (shaded) gives the actual standard deviation of the treatment effect estimates (i.e. the empirical 'truth'), both with 95% confidence intervals.

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.49 (0.48, 0.49)	0.45 (0.45, 0.45)	0.45 (0.45, 0.45)
		0.48 (0.47, 0.49)	0.45 (0.44, 0.46)	0.46 (0.45, 0.47)
Right PS Model				
	BOOM, BC	0.47 (0.47, 0.47)	0.44 (0.44, 0.44)	0.44 (0.44, 0.44)
		0.46 (0.45, 0.47)	0.46 (0.45, 0.47)	0.46 (0.45, 0.47)
	BOOM, no BC	0.50 (0.50, 0.50)	0.49 (0.49, 0.49)	0.49 (0.49, 0.49)
		0.46 (0.45, 0.47)	0.46 (0.45, 0.47)	0.46 (0.45, 0.47)
	OOM	0.65 (0.65, 0.65)	0.63 (0.63, 0.63)	0.64 (0.63, 0.64)
		0.64 (0.63, 0.66)	0.64 (0.63, 0.66)	0.64 (0.63, 0.65)
	IPW	0.53 (0.52, 0.53)	0.43 (0.43, 0.43)	0.45 (0.44, 0.45)
		0.48 (0.47, 0.49)	0.46 (0.45, 0.47)	0.47 (0.46, 0.48)
Wrong PS Model				
	BOOM, BC	0.47 (0.47, 0.47)	0.45 (0.45, 0.45)	0.44 (0.44, 0.45)
		0.46 (0.45, 0.47)	0.47 (0.46, 0.48)	0.46 (0.45, 0.47)
	BOOM, no BC	0.50 (0.50, 0.50)	0.50 (0.50, 0.50)	0.49 (0.49, 0.49)
		0.46 (0.45, 0.47)	0.47 (0.46, 0.48)	0.46 (0.45, 0.47)
	OOM	0.65 (0.64, 0.65)	0.64 (0.64, 0.64)	0.63 (0.63, 0.63)
		0.63 (0.62, 0.65)	0.63 (0.62, 0.65)	0.62 (0.61, 0.64)
	IPW	0.50 (0.50, 0.50)	0.44 (0.43, 0.44)	0.43 (0.43, 0.43)
		0.46 (0.45, 0.47)	0.47 (0.46, 0.48)	0.46 (0.45, 0.47)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.12: 10% treated: Accuracy of standard error estimates. For each estimator, the first row gives the mean standard error estimate and the second row (shaded) gives the actual standard deviation of the treatment effect estimates (i.e. the empirical 'truth'), both with 95% confidence intervals.

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.35 (0.35, 0.35)	0.33 (0.33, 0.33)	0.33 (0.32, 0.33)
		0.34 (0.34, 0.35)	0.32 (0.32, 0.33)	0.32 (0.32, 0.33)
Right PS Model				
	BOOM, BC	0.34 (0.34, 0.34)	0.32 (0.32, 0.32)	0.32 (0.32, 0.32)
		0.33 (0.32, 0.33)	0.33 (0.32, 0.33)	0.33 (0.32, 0.33)
	BOOM, no BC	0.36 (0.36, 0.36)	0.35 (0.35, 0.35)	0.35 (0.35, 0.35)
		0.33 (0.32, 0.33)	0.33 (0.32, 0.33)	0.33 (0.32, 0.33)
	OOM	0.45 (0.45, 0.45)	0.43 (0.43, 0.44)	0.44 (0.44, 0.44)
		0.44 (0.43, 0.45)	0.44 (0.43, 0.45)	0.44 (0.43, 0.44)
	IPW	0.39 (0.39, 0.39)	0.33 (0.32, 0.33)	0.34 (0.34, 0.34)
		0.36 (0.35, 0.36)	0.34 (0.33, 0.34)	0.34 (0.33, 0.35)
Wrong PS Model				
	BOOM, BC	0.34 (0.34, 0.34)	0.32 (0.32, 0.32)	0.32 (0.32, 0.32)
		0.33 (0.32, 0.33)	0.33 (0.33, 0.34)	0.33 (0.32, 0.33)
	BOOM, no BC	0.36 (0.36, 0.36)	0.36 (0.36, 0.36)	0.35 (0.35, 0.35)
		0.33 (0.32, 0.33)	0.33 (0.33, 0.34)	0.33 (0.32, 0.33)
	OOM	0.45 (0.45, 0.45)	0.44 (0.44, 0.44)	0.43 (0.43, 0.44)
		0.44 (0.43, 0.45)	0.43 (0.43, 0.44)	0.43 (0.42, 0.44)
	IPW	0.36 (0.36, 0.36)	0.33 (0.33, 0.33)	0.32 (0.32, 0.32)
		0.33 (0.32, 0.34)	0.33 (0.33, 0.34)	0.33 (0.32, 0.33)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.13: 20% treated: Accuracy of standard error estimates. For each estimator, the first row gives the mean standard error estimate and the second row (shaded) gives the actual standard deviation of the treatment effect estimates (i.e. the empirical 'truth'), both with 95% confidence intervals.

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.26 (0.26, 0.26)	0.25 (0.25, 0.25)	0.25 (0.25, 0.25)
		0.26 (0.26, 0.27)	0.25 (0.25, 0.26)	0.25 (0.25, 0.26)
Right PS Model				
	BOOM, BC	0.25 (0.25, 0.25)	0.24 (0.24, 0.24)	0.24 (0.24, 0.24)
		0.25 (0.25, 0.26)	0.25 (0.25, 0.26)	0.25 (0.25, 0.26)
	BOOM, no BC	0.26 (0.26, 0.27)	0.26 (0.26, 0.26)	0.26 (0.26, 0.26)
		0.25 (0.25, 0.26)	0.25 (0.25, 0.26)	0.25 (0.25, 0.26)
	OOM	0.32 (0.32, 0.32)	0.31 (0.31, 0.31)	0.31 (0.31, 0.31)
		0.32 (0.31, 0.32)	0.31 (0.31, 0.32)	0.31 (0.31, 0.32)
	IPW	0.30 (0.30, 0.30)	0.26 (0.25, 0.26)	0.26 (0.26, 0.26)
		0.28 (0.27, 0.29)	0.27 (0.26, 0.27)	0.27 (0.26, 0.27)
Wrong PS Model				
	BOOM, BC	0.25 (0.25, 0.25)	0.24 (0.24, 0.24)	0.24 (0.24, 0.24)
		0.25 (0.25, 0.26)	0.25 (0.25, 0.26)	0.25 (0.25, 0.26)
	BOOM, no BC	0.26 (0.26, 0.26)	0.26 (0.26, 0.27)	0.26 (0.26, 0.26)
		0.25 (0.25, 0.26)	0.25 (0.25, 0.26)	0.25 (0.25, 0.26)
	OOM	0.32 (0.32, 0.32)	0.31 (0.31, 0.31)	0.31 (0.31, 0.31)
		0.32 (0.31, 0.33)	0.32 (0.31, 0.32)	0.31 (0.31, 0.32)
	IPW	0.27 (0.27, 0.27)	0.25 (0.25, 0.25)	0.25 (0.25, 0.25)
		0.26 (0.25, 0.26)	0.26 (0.25, 0.26)	0.25 (0.25, 0.26)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

4.9.7 Tables for all prevalence levels: Coverage of nominal 95% confidence intervals

Table 4.14: 5% treated: Coverage of nominal 95% confidence intervals, with 95% Wilson confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.63 (0.61, 0.64)	0.95 (0.94, 0.96)	0.88 (0.88, 0.89)
Right PS Model				
	BOOM, BC	0.95 (0.94, 0.96)	0.94 (0.93, 0.94)	0.94 (0.93, 0.94)
	BOOM, no BC	0.96 (0.96, 0.97)	0.96 (0.95, 0.96)	0.96 (0.95, 0.96)
	OOM	0.95 (0.95, 0.96)	0.95 (0.94, 0.95)	0.95 (0.95, 0.96)
	IPW	0.96 (0.96, 0.97)	0.93 (0.92, 0.93)	0.94 (0.93, 0.94)
Wrong PS Model				
	BOOM, BC	0.94 (0.93, 0.95)	0.94 (0.93, 0.94)	0.92 (0.92, 0.93)
	BOOM, no BC	0.95 (0.95, 0.96)	0.96 (0.95, 0.96)	0.95 (0.94, 0.96)
	OOM	0.95 (0.94, 0.95)	0.96 (0.95, 0.96)	0.94 (0.94, 0.95)
	IPW	0.94 (0.93, 0.94)	0.93 (0.92, 0.93)	0.90 (0.89, 0.91)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.15: 10% treated: Coverage of nominal 95% confidence intervals, with 95% Wilson confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.48 (0.47, 0.50)	0.95 (0.94, 0.95)	0.87 (0.86, 0.88)
Right PS Model				
	BOOM, BC	0.95 (0.94, 0.96)	0.94 (0.93, 0.94)	0.94 (0.93, 0.95)
	BOOM, no BC	0.96 (0.96, 0.97)	0.96 (0.95, 0.96)	0.96 (0.95, 0.97)
	OOM	0.96 (0.95, 0.96)	0.95 (0.94, 0.95)	0.95 (0.95, 0.96)
	IPW	0.97 (0.96, 0.97)	0.94 (0.93, 0.94)	0.94 (0.94, 0.95)
Wrong PS Model				
	BOOM, BC	0.93 (0.92, 0.93)	0.94 (0.93, 0.94)	0.91 (0.91, 0.92)
	BOOM, no BC	0.94 (0.93, 0.95)	0.96 (0.95, 0.96)	0.94 (0.93, 0.95)
	OOM	0.94 (0.94, 0.95)	0.95 (0.95, 0.96)	0.94 (0.93, 0.94)
	IPW	0.92 (0.91, 0.92)	0.94 (0.93, 0.94)	0.90 (0.89, 0.91)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

Table 4.16: 20% treated: Coverage of nominal 95% confidence intervals, with 95% Wilson confidence intervals (showing Monte Carlo uncertainty).

		Outcome Model		
	Estimator	None	Right	Wrong
No PS Model				
	OLS	0.39 (0.38, 0.41)	0.95 (0.94, 0.95)	0.86 (0.85, 0.87)
Right PS Model				
	BOOM, BC	0.95 (0.94, 0.95)	0.94 (0.93, 0.94)	0.94 (0.93, 0.94)
	BOOM, no BC	0.96 (0.95, 0.96)	0.96 (0.95, 0.96)	0.96 (0.95, 0.96)
	OOM	0.96 (0.95, 0.96)	0.95 (0.94, 0.95)	0.95 (0.95, 0.96)
	IPW	0.97 (0.96, 0.97)	0.94 (0.93, 0.95)	0.95 (0.94, 0.95)
Wrong PS Model				
	BOOM, BC	0.91 (0.90, 0.92)	0.94 (0.93, 0.95)	0.90 (0.90, 0.91)
	BOOM, no BC	0.93 (0.92, 0.93)	0.96 (0.95, 0.96)	0.93 (0.92, 0.94)
	OOM	0.93 (0.93, 0.94)	0.95 (0.94, 0.95)	0.93 (0.92, 0.93)
	IPW	0.89 (0.88, 0.90)	0.94 (0.94, 0.95)	0.89 (0.88, 0.89)

PS: propensity score. OLS: ordinary least squares. BOOM: bagged one-to-one matching. BC: bias correction. OOM: one-to-one matching. IPW: inverse probability of treatment weighting.

4.10 Appendix C. Data-preparation code from case study

```
# (mydat1 is the raw RHC data, imported from web)
mydat1 <- within(mydat1, {
    swang1.01 <- ifelse(swang1 == "RHC", 1, 0)
    days.in.hosp <- dschdte - sadmdte
    days.alive <- dthdte - sadmdte
    los <- pmin(days.in.hosp, days.alive, na.rm= TRUE)

    # messing up latex
    ninsclas <- gsub("&", "+", ninsclas, fixed= TRUE)
})

mydat2 <- mydat1[!is.na(mydat1$dschdte), ]

mydat <- mydat2

mydat <- within(mydat, {
    # if no secondary disease category, set to "None listed"
    cat2 <- ifelse(is.na(cat2), "None Listed", cat2)
})
```

4.11 Appendix D. Formulas from case study

```
# propensity score model
ps.form <- swang1 ~
    ninsclas +
    cat1 +
    cat2 +
    neuro +
    dnr1 +
    surv2md1 +
    scoma1 +
    meanbp1 +
    hema1 +
    sod1 +
    renalhx +
    gibledhx

# outcome model
lm.form <- update.formula(ps.form, los ~ .)
lm.form <- update.formula(lm.form, . ~ . + swang1.01)
```

Chapter 5

Conclusion

This dissertation has examined several topics related to the estimation of the average effect of treatment on the evenly matchable units (ATM). Chapter 2 defined the ATM, summarized existing techniques that can be used to estimate it, introduced new ATM estimation techniques, and illustrated the use of some of these techniques in a case study. Chapter 3 introduced Visual Pruner, a new web application that can serve as a first step in estimation of the ATM. Chapter 4 introduced bagged one-to-one matching (BOOM), a new technique that can be used to estimate the ATM directly or to generate weights for use in ATM estimation, as discussed in Chapter 2.

Although I believe that this work presents the first formal definition of the ATM, researchers have in essence been estimating the ATM for years, without explicitly naming it as such. As Li and Greene (2013) show, the popular propensity-score pair matching estimator estimates what we now call the ATM unless there are at least as many control units as treatment units throughout the range of propensity scores, in which case the ATM is equivalent to the ATT. Thus although researchers who have conducted PS caliper pair matching in the past may have wanted to estimate the ATT, their estimates may have more accurately reflected the ATM. Furthermore, as Rosenbaum (2012) and Li and Greene (2013) argue, a quantity such as the one we now call the ATM may in fact be of greater interest than the ATT in many situations. Because the questions of which estimands are of interest and which quantities are actually estimable in a given dataset are fundamental questions in observational studies, I hope that explicit consideration of the ATM as a possible target of inference will bring added clarity to other researchers engaged in this work.

While the three papers presented here provide a foundation for inference about the ATM, both Visual Pruner and BOOM can be used when estimands other than the ATM are of interest, and even in cases where the researcher is not explicitly interested in estimating a causal effect. Because Visual Pruner allows transparent and reproducible selection of a study cohort from the available data, use of the app can improve both the generalizability and the credibility of observational studies. BOOM is a highly flexible technique that shows great promise as a tool for reducing both bias and variance in effect estimation, and the weights generated by the BOOM process essentially allow the conversion of any matching process into a weighting system with all of the benefits of weighting. Both of these tools are freely available

on the web, and I look forward to incorporating the planned extensions discussed in Chapters 3 and 4 as well as suggestions from users of these tools in the future.

REFERENCES

Althauser, R. P. and Rubin, D. (1970), The Computerized Construction of a Matched Sample, *American Journal of Sociology* **76**(2), 325–346.

Austin, P. C. (2009*a*), Balance Diagnostics for Comparing the Distribution of Baseline Covariates between Treatment Groups in Propensity-Score Matched Samples, *Statistics in Medicine* **28**(25), 3083–3107.

Austin, P. C. (2009*b*), Using the standardized difference to compare the prevalence of a binary variable between two groups in observational research, *Communications in Statistics-Simulation and ...* **38**, 1228–1234.

Austin, P. C. (2011), An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies, *Multivariate Behavioral Research* **46**(3), 399–424.

Austin, P. C. (2013), The performance of different propensity score methods for estimating marginal hazard ratios., *Statistics in Medicine* **32**(16), 2837–2849.

Austin, P. C. and Small, D. S. (2014), The Use of Bootstrapping When Using Propensity-Score Matching without Replacement: A Simulation Study, *Statistics in Medicine* .

Austin, P. C. and Stuart, E. A. (2015), Moving Towards Best Practice When Using Inverse Probability of Treatment Weighting (IPTW) Using the Propensity Score to Estimate Causal Treatment Effects in Observational Studies, *Statistics in Medicine* **34**(28), 3661–3679.

Binder, D. A. (1983), On the Variances of Asymptotically Normal Estimators from Complex Surveys, *International Statistical Review/Revue Internationale de Statistique* **51**(3), 279–292.

Bishop, Y. M., Fienberg, S. E. and Holland, P. W. (1975), *Discrete Multivariate Analysis: Theory and Practice*, MIT Press.

Breiman, L. (1996), Bagging Predictors, *Machine learning* **24**(2), 123–140.

Breiman, L. (2001), Random forests, *Machine learning* **45**(1), 5–32.

Cham, H. and West, S. G. (2016), Propensity Score Analysis with Missing Data, *Psychological methods* **21**(3), 427–445.

Chang, W., Cheng, J., Allaire, J., Xie, Y. and McPherson, J. (2015), ***shiny**: Web Application Framework for R*. R package version 0.12.0.9002.
URL: *http://shiny.rstudio.com*

Cole, S. R. and Hernán, M. A. (2008), Constructing Inverse Probability Weights for Marginal Structural Models, *American Journal of Epidemiology* **168**(6), 656–664.

Connors, A. F., Speroff, T., Dawson, N. V., Thomas, C., Harrell, F. E., Wagner, D., Desbiens, N., Goldman, L., Wu, A. W., Califf, R. M., Fulkerson, W. J., Vidaillet, H., Broste, S., Bellamy, P., Lynn, J. and Knaus, W. A. (1996), The effectiveness of right heart catheterization in the initial care of critically ill patients, *JAMA* **276**(11), 889–897.

Crump, R. K., Hotz, V. J., Imbens, G. W. and Mitnik, O. A. (2009), Dealing with Limited Overlap in Estimation of Average Treatment Effects, *Biometrika* **96**(1), 187–199.

Diamond, A. and Sekhon, J. S. (2013), Genetic matching for estimating causal effects: A general multivariate matching method for achieving balance in observational studies, *Review of Economics and Statistics* **95**(3), 932–945.

Dowle, M., Srinivasan, A., Short, T., with contributions from R Saporta, S. L. and Antonyan, E. (2015), ***data.table**: Extension of Data.frame*. R package version 1.9.6.
URL: *https://CRAN.R-project.org/package=data.table*

Efron, B. (2014), Estimation and Accuracy After Model Selection, *Journal of the American Statistical Association* **109**(507), 991–1007.

Flury, B. K. and Riedwyl, H. (1986), Standard distance in univariate and multivariate analysis, *The American Statistician* **40**(3), 249–251.

Greenland, S., Robins, J. M. and Pearl, J. (1999), Confounding and collapsibility in causal inference, *Statistical Science* **14**(1), 29–46.

Greevy, R. A., Grijalva, C. G., Roumie, C. L., Beck, C., Hung, A. M., Murff, H. J., Liu, X. and Griffin, M. R. (2012), Reweighted Mahalanobis distance matching for cluster-randomized trials with missing data., *Pharmacoepidemiology and Drug Safety* **21 Suppl 2**, 148–154.

Grijalva, C. G., Roumie, C. L., Murff, H. J., Hung, A. M., Beck, C., Liu, X., Griffin, M. R. and Greevy, R. A. (2015), The role of matching when adjusting for baseline differences in the outcome variable of comparative effectiveness studies., *Journal of Comparative Effectiveness Research* **4 (4)**, 341–349.

Hansen, B. B. (2008), The Prognostic Analogue of the Propensity Score, *Biometrika* **95**(2), 481–488.

Harrell, F. E. (2015), *Regression Modeling Strategies: With Applications to Linear Models, Logistic and Ordinal Regression, and Survival Analysis (Springer Series in Statistics)*, 2nd edn, Springer.

Harrell Jr., F. E. (2015), ***rms****: Regression Modeling Strategies*. R package version 4.3-1.
URL: *http://CRAN.R-project.org/package=rms*

Hesterberg, T. (2014), What Teachers Should Know about the Bootstrap: Resampling in the Undergraduate Statistics Curriculum, *arXiv.org* .

Ho, D. E., Imai, K., King, G. and Stuart, E. A. (2007), Matching As Nonparametric Preprocessing for Reducing Model Dependence in Parametric Causal Inference, *Political analysis* **15**(3), 199–236.

Iacus, S. M., King, G. and Porro, G. (2012), Causal Inference without Balance Checking: Coarsened Exact Matching, *Political analysis* **20**(1), 1–24.

Imbens, G. W. (2004), Nonparametric estimation of average treatment effects under exogeneity: A review, *Review of Economics and Statistics* **86**(1), 4–29.

Kang, J. D. Y. and Schafer, J. L. (2007), Demystifying Double Robustness: A Comparison of Alternative Strategies for Estimating a Population Mean from Incomplete Data, *Statistical Science* **22**(4), 523–539.

King, G. and Zeng, L. (2006), The Dangers of Extreme Counterfactuals, *Political analysis* **14**(2), 131–159.

King, G. and Zeng, L. (2007), When Can History be Our Guide? The Pitfalls of Counterfactual Inference, *International Studies Quarterly* **51**(1), 183–210.

Lee, B. K., Lessler, J. and Stuart, E. A. (2010), Improving propensity score weighting using machine learning., *Statistics in Medicine* **29**(3), 337–346.

Li, L. and Greene, T. (2013), A Weighting Analogue to Pair Matching in Propensity Score Analysis, *The International Journal of Biostatistics* **9**(2), 215–234.

Lumley, T. (2004), Analysis of complex survey samples, *Journal of Statistical Software* **9**(1), 1–19. R package version 2.2.

Lumley, T. (2011), *Complex Surveys: A Guide to Analysis Using R (Wiley Series in Survey Methodology)*, Wiley.

Lumley, T. (2014), '**survey**: Analysis of complex survey samples'. R package version 3.30.

Neyman, J., Iwaszkiewicz, K. and Kolodziejczyk, S. (1935), Statistical Problems in Agricultural Experimentation, *Supplement to the Journal of the Royal Statistical Society* **2**(2), 107–180.

Porro, G. and Iacus, S. M. (2009), Random recursive partitioning: A matching method for the estimation of the average treatment effect, *Journal of Applied Econometrics* **24**(1), 163–185.

R Core Team (2015), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: *http://www.R-project.org/*

Rosenbaum, P. R. (1987), Model-Based Direct Adjustment, *Journal of the American Statistical Association* **82**(398), 387–394.

Rosenbaum, P. R. (1991), A characterization of optimal designs for observational studies, *Journal of the Royal Statistical Society Series B* **53**(3), 597–610.

Rosenbaum, P. R. (2010), *Design of Observational Studies (Springer Series in Statistics)*, Springer.

Rosenbaum, P. R. (2012), Optimal Matching of an Optimally Chosen Subset in Observational Studies, *Journal of Computational and Graphical Statistics* **21**(1), 57–71.

Rosenbaum, P. R. and Rubin, D. B. (1983), The Central Role of the Propensity Score in Observational Studies for Causal Effects, *Biometrika* **70**(1), 41–55.

Rosenbaum, P. R. and Rubin, D. B. (1984), Reducing Bias in Observational Studies Using Subclassification on the Propensity Score, *Journal of the American Statistical Association* **79**(387), 516–524.

Rubin, D. B. (1974), Estimating causal effects of treatments in randomized and non-randomized studies., *Journal of educational Psychology* **66**(5), 688–701.

Rubin, D. B. (1980), Bias reduction using Mahalanobis-metric matching, *Biometrics* **36**, 293–298.

Samuels, L. R. and Greevy, Jr., R. A. (2016*a*), Bagged One-to-One Matching for Efficient and Robust Treatment Effect Estimation. In preparation.

Samuels, L. R. and Greevy, Jr., R. A. (2016*b*), The Average Treatment Effect on the Evenly Matchable Units (ATM): A Valuable Estimand in Causal Inference. In preparation.

Schafer, J. L. and Kang, J. (2008), Average Causal Effects from Nonrandomized Studies: A Practical Guide and Simulated Example, *Psychological methods* **13**(4), 279–313.

Sekhon, J. S. (2011), Multivariate and propensity score matching software with automated balance optimization: The Matching package for R, *Journal of Statistical Software* **42**(7), 1–52.
URL: *http://www.jstatsoft.org/v42/i07/*

Setoguchi, S., Schneeweiss, S., Brookhart, M. A., Glynn, R. J. and Cook, E. F. (2008), Evaluating uses of data mining techniques in propensity score estimation: a simulation study, *Pharmacoepidemiology and Drug Safety* **17**(6), 546–555.

Sexton, J. and Laake, P. (2009), Standard Errors for Bagged and Random Forest Estimators, *Computational Statistics and Data Analysis* **53**, 801–811.

Stuart, E. A. (2010), Matching Methods for Causal Inference: A Review and a Look Forward, *Statistical Science* **25**(1), 1–21.

Traskin, M. and Small, D. S. (2011), Defining the Study Population for an Observational Study to Ensure Sufficient Overlap: A Tree Approach, *Statistics in Biosciences* **3**(1), 94–118.

Waernbaum, I. (2012), Model misspecification and robustness in causal inference: comparing matching with doubly robust estimation., *Statistics in Medicine* **31**(15), 1572–1581.

Wager, S., Hastie, T. and Efron, B. (2014), Confidence Intervals for Random Forests: The Jackknife and the Infinitesimal Jackknife, *Journal of Machine Learning Research* **15**(1), 1625–1651.

Yang, D. and Dalton, J. E. (2012), A unified approach to measuring the effect size between two groups using SAS®, *SAS Global Forum* .

Yoshida, K. and Bohn, J. (2015), ***tableone**: Create 'Table 1' to Describe Baseline Characteristics*. R package version 0.7.3.
URL: *https://CRAN.R-project.org/package=tableone*