Biometrika (2021), **108**, 2, *pp.* 299–319
Printed in Great Britain
doi: 10.1093/biomet/asaa076
Advance Access publication 28 December 2020

Quasi-oracle estimation of heterogeneous treatment effects

By X. NIE

Department of Computer Science, Stanford University,
353 Serra Mall, Stanford, California 94305, U.S.A.
xinkun@stanford.edu

and S. WAGER

Stanford Graduate School of Business, 655 Knight Way, Stanford, California 94305, U.S.A.
swager@stanford.edu

Summary

Flexible estimation of heterogeneous treatment effects lies at the heart of many statistical applications, such as personalized medicine and optimal resource allocation. In this article we develop a general class of two-step algorithms for heterogeneous treatment effect estimation in observational studies. First, we estimate marginal effects and treatment propensities to form an objective function that isolates the causal component of the signal. Then, we optimize this data-adaptive objective function. The proposed approach has several advantages over existing methods. From a practical perspective, our method is flexible and easy to use: in both steps, any loss-minimization method can be employed, such as penalized regression, deep neural networks, or boosting; moreover, these methods can be fine-tuned by cross-validation. Meanwhile, in the case of penalized kernel regression, we show that our method has a quasi-oracle property. Even when the pilot estimates for marginal effects and treatment propensities are not particularly accurate, we achieve the same error bounds as an oracle with prior knowledge of these two nuisance components. We implement variants of our approach based on penalized regression, kernel ridge regression, and boosting in a variety of simulation set-ups, and observe promising performance relative to existing baselines.

Some key words: Boosting; Causal inference; Empirical risk minimization; Kernel regression; Penalized regression.

1. Introduction

The problem of heterogeneous treatment effect estimation in observational studies arises in a wide variety of application areas (Athey, 2017), ranging from personalized medicine (Obermeyer & Emanuel, 2016) to offline evaluation of bandits (Dudík et al., 2011), and is also a key component of several proposals for learning decision rules (Hirano & Porter, 2009; Athey & Wager, 2019). There has been considerable interest in developing flexible and well-performing methods for heterogeneous treatment effect estimation. Some notable recent advances include proposals based on the lasso (Imai & Ratkovic, 2013), recursive partitioning (Su et al., 2009; Athey & Imbens, 2016), Bayesian additive regression trees (Hill, 2011; Hahn et al., 2020), random forests (Wager & Athey, 2018), boosting (Powers et al., 2018), neural networks (Shalit et al., 2017), and combinations thereof (Künzel et al., 2019); see Dorie et al. (2019) for a recent survey and comparisons.

Although this line of work has led to many promising developments, the literature has not yet settled on a comprehensive answer as to how machine learning methods should be adapted for treatment effect estimation in observational studies. The process of developing causal variants of machine learning methods is in practice a labour-intensive process, requiring the involvement of specialized researchers. Moreover, with some exceptions, the above methods have mostly been justified via numerical experiments and come with no formal convergence guarantees or error bounds proving that the methods actually succeed in isolating causal effects better than a simple nonparametric regression-based approach would.

In this article we propose a new approach to estimating heterogeneous treatment effects that addresses both of the above concerns. Our framework allows for fully automatic specification of heterogeneous treatment effect estimators in terms of arbitrary loss-minimization procedures. Moreover, we show how the resulting methods can achieve error bounds comparable to those of oracle methods that know everything about the data-generating distribution except the treatment effects. Conceptually, our approach fits into a research programme, outlined by van der Laan & Dudoit (2003) and later developed by Luedtke & van der Laan (2016c), Chernozhukov et al. (2018) and other works cited therein, whereby ideas on doubly robust estimation are combined with oracle inequalities and cross-validation to develop loss functions that can be used for principled statistical estimation using generic machine learning tools.

2. A LOSS FUNCTION FOR TREATMENT EFFECT ESTIMATION

We formalize the problem in terms of the potential outcomes framework (Neyman, 1923; Rubin, 1974). The analyst has access to n independent and identically distributed examples (X_i, Y_i, W_i) $(i = 1, \ldots, n)$, where $X_i \in \mathcal{X}$ denotes per-person features, $Y_i \in \mathbb{R}$ is the observed outcome, and $W_i \in \{0, 1\}$ is the treatment assignment. We posit the existence of potential outcomes $\{Y_i(0), Y_i(1)\}$, which correspond to the outcomes that would have been observed given the treatment assignments $W_i = 0$ and $W_i = 1$, respectively, such that $Y_i = Y_i(W_i)$, and seek to estimate the conditional average treatment effect function $\tau^*(x) = E\{Y(1) - Y(0) \mid X = x\}$. In order to identify $\tau^*(x)$, we assume unconfoundedness, i.e., that the treatment assignment is randomized once the features X_i are controlled for (Rosenbaum & Rubin, 1983).

Assumption 1. The treatment assignment W_i is unconfounded, i.e., $\{Y_i(0),\ Y_i(1)\} \perp\!\!\!\perp W_i \mid X_i$.

We write the treatment propensity as $e^*(x) = \text{pr}(W = 1 \mid X = x)$ and the conditional response surfaces as $\mu^*_{(w)}(x) = E\{Y(w) \mid X = x\}$ for $w \in \{0, 1\}$; throughout this paper we use $*$ superscripts to denote unknown population quantities. Then, under unconfoundedness,

$$E\{\varepsilon_i(W_i) \mid X_i,\ W_i\} = 0, \quad \varepsilon_i(w) := Y_i(w) - \{\mu^*_{(0)}(X_i) + w\tau^*(X_i)\}.$$

Given this set-up, it is helpful to rewrite the conditional average treatment effect function $\tau^*(x)$ in terms of the conditional mean outcome $m^*(x) = E(Y \mid X = x) = \mu^*_{(0)}(X_i) + e^*(X_i)\tau^*(X_i)$ as follows, with the shorthand $\varepsilon_i = \varepsilon_i(W_i)$:

$$Y_i - m^*(X_i) = \{W_i - e^*(X_i)\}\,\tau^*(X_i) + \varepsilon_i. \tag{1}$$

This decomposition was originally used by Robinson (1988) to estimate parametric components in partially linear models, and has received considerable attention in recent years. Athey et al. (2019) rely on it to grow a causal forest that is robust to confounding; Robins (2004) builds

on it to develop G-estimation for sequential trials; and Chernozhukov et al. (2018) use it as a leading example of how machine learning methods can be put to good use in estimating nuisance components for semiparametric inference. All these results, however, concern the estimation of parametric models for $\tau(\cdot)$ or, in the case of Athey et al. (2019), local parametric modelling.

The goal of the present paper is to study how we can use the Robinson transformation (1) for flexible treatment effect estimation that builds on modern machine learning approaches such as boosting or deep learning. Our main result is that this representation can be used to construct a loss function that captures heterogeneous treatment effects, and we can then estimate treatment effects accurately, in terms of both empirical performance and asymptotic guarantees, by finding regularized minimizers of this loss function.

As motivation for our approach, (1) can equivalently be expressed as (Robins, 2004)

$$\tau^*(\cdot) = \arg\min_{\tau} \left\{ E\left(\left[\{Y_i - m^*(X_i)\} - \{W_i - e^*(X_i)\}\, \tau(X_i) \right]^2 \right) \right\}, \tag{2}$$

and so an oracle that knew both of the functions $m^*(x)$ and $e^*(x)$ a priori could estimate the heterogeneous treatment effect function $\tau^*(\cdot)$ by empirical loss minimization,

$$\tilde{\tau}(\cdot) = \arg\min_{\tau} \left(\frac{1}{n} \sum_{i=1}^{n} \left[\{Y_i - m^*(X_i)\} - \{W_i - e^*(X_i)\}\, \tau(X_i) \right]^2 + \Lambda_n\{\tau(\cdot)\} \right), \tag{3}$$

where the term $\Lambda_n\{\tau(\cdot)\}$ is interpreted as a regularizer on the complexity of the $\tau(\cdot)$ function. This regularization could be explicit, as in penalized regression, or implicit, as provided by a carefully designed deep neural network, for instance. The difficulty, however, is that in practice we never know the weighted main effect function $m^*(x)$ and usually do not know the treatment propensities $e^*(x)$ either, and so the estimator (3) is not feasible.

With these preliminaries, we study the following class of two-step estimators using cross-fitting (Schick, 1986; Chernozhukov et al., 2018) motivated by the above oracle procedure.

Step 1. Divide up the data into Q evenly sized folds, where Q is typically set to 5 or 10. Let $q(\cdot)$ be a mapping from the sample indices $i = 1, \ldots, n$ to the Q evenly sized data folds, and fit $\hat{m}$ and $\hat{e}$ with cross-fitting over the Q folds using methods tuned for optimal predictive accuracy.

Step 2. Estimate treatment effects via a plug-in version of (3), where the $\hat{e}^{\{-q(i)\}}(X_i)$ and $\hat{m}^{\{-q(i)\}}(X_i)$ denote predictions made without using the data fold that the ith training example belongs to:

$$\begin{aligned} \hat{\tau}(\cdot) &= \arg\min_{\tau} \left[\hat{L}_n\{\tau(\cdot)\} + \Lambda_n\{\tau(\cdot)\} \right], \\ \hat{L}_n\{\tau(\cdot)\} &= \frac{1}{n} \sum_{i=1}^{n} \left[\{Y_i - \hat{m}^{\{-q(i)\}}(X_i)\} - \{W_i - \hat{e}^{\{-q(i)\}}(X_i)\}\, \tau(X_i) \right]^2. \end{aligned} \tag{4}$$

In other words, the first step learns an approximation for the oracle objective, and the second step optimizes it. We refer to this approach as the R-learner, in recognition of the work of Robinson (1988) and to emphasize the role of residualization. We will also refer to the squared loss $\hat{L}_n\{\tau(\cdot)\}$ as the R-loss.

This paper makes the following contributions. First, we implement variants of our method based on penalized regression, kernel ridge regression, and boosting. In each case, we find that

the R-learner exhibits promising performance relative to existing methods. Second, we prove that in the case of penalized kernel regression, error bounds for the feasible estimator of $\hat{\tau}(\cdot)$ asymptotically match the best available bounds for the oracle method $\tilde{\tau}(\cdot)$. The main point here is that, heuristically, the rate of convergence of $\hat{\tau}(\cdot)$ depends only on the functional complexity of $\tau^*(\cdot)$, but not on the functional complexity of $m^*(\cdot)$ and $e^*(\cdot)$. More formally, provided we estimate $m^*(\cdot)$ and $e^*(\cdot)$ at $o(n^{-1/4})$ rates in root-mean-squared error, we can achieve considerably faster rates of convergence for $\hat{\tau}(\cdot)$, and these rates depend only on the complexity of $\tau^*(\cdot)$. The oracle version (2) of our loss function is a member of a class of loss functions for heterogeneous treatment effect estimation considered in Luedtke & van der Laan (2016c), and the results in that paper immediately imply large-sample consistency of the minimizer of this oracle loss. Our contribution is the result on rates, specifically that the estimation error in nuisance components does not affect the excess loss bounds for $\hat{\tau}(\cdot)$.

The R-learning approach has several practical advantages over existing, more ad hoc methods. Any good heterogeneous treatment effect estimator needs to achieve two goals: first, it should eliminate spurious effects by controlling for correlations between $e^*(X)$ and $m^*(X)$; second, it should accurately express $\tau^*(\cdot)$. Most existing machine learning approaches to treatment effect estimation seek to provide an algorithm that accomplishes both tasks at once (see, e.g., Shalit et al., 2017; Powers et al., 2018; Wager & Athey, 2018). In contrast, the R-learner cleanly separates these two tasks: we eliminate spurious correlations via the structure of the loss function $\hat{L}_n$, and we can induce a representation for $\hat{\tau}(\cdot)$ by choosing the method by which we optimize (4).

This separation of tasks allows for considerable algorithmic flexibility: optimizing (4) is an empirical minimization problem, and so can be efficiently solved using off-the-shelf software such as `glmnet` for high-dimensional regression (Friedman et al., 2010), `XGboost` for boosting (Chen & Guestrin, 2016), or `TensorFlow` for deep learning (Abadi et al., 2016). Furthermore, we can tune any of these methods by cross-validating on the loss $\hat{L}_n$, which avoids the use of more sophisticated model-assisted cross-validation procedures such as those developed in Athey & Imbens (2016) and Powers et al. (2018). Relatedly, the machine learning method used to optimize (4) only needs to find a generalizable minimizer of $\hat{L}_n$, rather than also control for spurious correlations, and thus we can confidently use black-box methods without auditing their internal state to check that they properly control for confounding; instead, we only need to verify that the methods do in fact find good minimizers of $\hat{L}_n$ on hold-out data.

3. RELATED WORK

Under unconfoundedness, Assumption 1, the conditional average treatment effect function can be written as $\tau^*(x) = \mu^*_{(1)}(x) - \mu^*_{(0)}(x)$ with $\mu^*_{(w)}(x) = E(Y \mid X = x,\ W = w)$. As a consequence of this representation, it may be tempting to first estimate $\hat{\mu}_{(w)}(x)$ on the treated and control samples separately and then set $\hat{\tau}(x) = \hat{\mu}_{(1)}(x) - \hat{\mu}_{(0)}(x)$. This approach, however, is often not robust: because $\hat{\mu}_{(1)}(x)$ and $\hat{\mu}_{(0)}(x)$ are not trained together, their difference may be unstable. As an example, consider fitting the lasso (Tibshirani, 1996) to estimate $\hat{\mu}_{(1)}(x)$ and $\hat{\mu}_{(0)}(x)$ in the high-dimensional linear model $Y_i(w) = X_i^{\mathrm{T}}\beta^*_{(w)} + \varepsilon_i(w)$ with $X_i, \beta^*_{(w)} \in \mathbb{R}^d$ and $E(\varepsilon_i(w) \mid X_i) = 0$. A naive way to do this would be to fit two separate lassos to the treated and control samples,

$$\hat{\beta}_{(w)} = \underset{\beta_{(w)}}{\arg\min} \left\{ \sum_{\{i:\, W_i = w\}} \left(Y_i - X_i^{\mathrm{T}}\beta_{(w)}\right)^2 + \lambda_{(w)} \left\|\beta_{(w)}\right\|_1 \right\}, \tag{5}$$

and then use them to deduce a treatment effect function, $\hat{\tau}(x) = x^{\mathrm{T}}(\hat{\beta}_{(1)} - \hat{\beta}_{(0)})$. However, the fact that $\hat{\beta}_{(0)}$ and $\hat{\beta}_{(1)}$ are regularized towards 0 separately may inadvertently lead to the treatment effect estimate $\hat{\beta}_{(1)} - \hat{\beta}_{(0)}$ being regularized away from 0, even when $\tau^*(x) = 0$ everywhere. This problem is especially acute when the treated and control samples are of different sizes; see Künzel et al. (2019) for some striking examples.

In the recent literature on heterogeneous treatment effect estimation, several ideas on how to avoid such regularization bias have been suggested. Some authors have proposed structural changes to various machine learning methods aimed at accurate estimation of $\tau(\cdot)$ (Su et al., 2009; Imai & Ratkovic, 2013; Athey & Imbens, 2016; Shalit et al., 2017; Powers et al., 2018; Wager & Athey, 2018; Hahn et al., 2020). For example, with the lasso, Imai & Ratkovic (2013) advocate replacing (5) with a single lasso as follows:

$$(\hat{b}, \hat{\delta}) = \underset{b,\delta}{\arg\min} \left[\sum_{i=1}^{n} \left\{ Y_i - X_i^{\mathrm{T}} b + (W_i - 0.5) X_i^{\mathrm{T}} \delta \right\}^2 + \lambda_b \|b\|_1 + \lambda_\delta \|\delta\|_1 \right], \tag{6}$$

where then $\hat{\tau}(x) = x^{\mathrm{T}}\hat{\delta}$. This approach always correctly regularizes towards a sparse δ-vector for treatment heterogeneity. The other methods cited above involve variants and improvements of similar ideas in the context of more sophisticated machine learning methods; see, for example, Shalit et al. (2017, Fig. 1) for a neural network architecture designed to highlight treatment effect heterogeneity without being affected by confounders.

Here, instead of trying to modify the algorithms underlying different machine learning tools to improve their performance as treatment effect estimators, we focus on modifying the loss function used to train generic machine learning methods. In doing so, we build on the research programme developed by van der Laan & Dudoit (2003), van der Laan & Rubin (2006) and van der Laan et al. (2007), and later fleshed out in the context of individualized treatment rules by Luedtke & van der Laan (2016a,b,c). In their report, van der Laan & Dudoit proposed choosing the best among a potentially growing set of generic statistical rules by cross-validating on a doubly robust objective. In the absence of nuisance components, an ε-net version of this procedure was shown to have good asymptotic properties (van der Laan et al., 2006). Meanwhile, Luedtke & van der Laan (2016c) considered a class of valid objectives for learning either individualized treatment rules or heterogeneous treatment effects, within which the oracle version (2) of our loss function fits, and discussed properties of model averaging and cross-validation with these objectives. Our contributions with respect to this line of work include using the R-loss for treatment effect estimation via generic machine learning and developing strong excess loss bounds $\hat{\tau}(\cdot)$ that hold for a computationally tractable and widely used approach to nonparametric estimation, namely penalized regression over a reproducing kernel Hilbert space.

Another related strand in the literature focuses on meta-learning approaches that are not closely tied to any specific machine learning method. Künzel et al. (2019) proposed two approaches to heterogeneous treatment effect estimation via generic machine learning methods. One, called the X-learner, first estimates $\hat{\mu}_{(w)}(x)$ with appropriate nonparametric regression methods; then, on the treated observations, it defines pseudo-effects $D_i = Y_i - \hat{\mu}_{(0)}^{(-i)}(X_i)$ and uses them to fit $\hat{\tau}_{(1)}(X_i)$ via a nonparametric regression. A second estimator, $\hat{\tau}_{(0)}(X_i)$, is obtained analogously, and the two treatment effect estimators are aggregated into

$$\hat{\tau}(x) = \{1 - \hat{e}(x)\}\, \hat{\tau}_{(1)}(x) + \hat{e}(x) \hat{\tau}_{(0)}(x). \tag{7}$$

Another method, called the U-learner, starts with the observation that

$$E(U_i \mid X_i = x) = \tau(x), \quad U_i = \frac{Y_i - m^*(X_i)}{W_i - e^*(X_i)},$$

and then fits U_i on X_i using any off-the-shelf method. Athey & Imbens (2016) and Tian et al. (2014) developed related methods for heterogeneous treatment effect estimation based on weighting the outcomes or the covariates with the propensity score; for example, one can estimate $\tau^*(\cdot)$ by regressing $Y_i\{W_i - e^*(X_i)\}/[e^*(X_i)\{1 - e^*(X_i)\}]$ on X_i. In our experiments, we compare our method at length to those of Künzel et al. (2019). Again, in regard to this line of work, our main contributions are the R-learner method, which yields meaningful improvements over baselines in a variety of settings, and our associated analysis, which provides a quasi-oracle error bound on the conditional average treatment effect function, i.e., where the error of $\hat{\tau}$ may decay faster than that of $\hat{e}$ or $\hat{m}$.

The closest result to ours is that of Zhao et al. (2018), which combines Robinson's transformation with the lasso to obtain valid post-selection inference on effect modification in the high-dimensional linear model. To the best of our knowledge, the present work is the first to use Robinson's transformation to motivate a loss function that is used in a general machine learning context.

Our formal results draw from the literature on semiparametric efficiency and construction of orthogonal moments, including Robinson (1988) and, more broadly, Bickel et al. (1993), Newey (1994),Robins & Rotnitzky (1995), Robins (2004), Tsiatis (2007), van der Laan & Rose (2011), Belloni et al. (2017), Robins et al. (2017) and Chernozhukov et al. (2018), among others, which aim at $\sqrt{n}$-rate estimation of a target parameter in the presence of nuisance components that cannot be estimated at a $\sqrt{n}$ rate. Algorithmically, our approach has a close connection to targeted maximum likelihood estimation (Scharfstein et al., 1999; van der Laan & Rubin, 2006), which starts by estimating nuisance components nonparametrically, and then uses these first-stage estimates to define a likelihood function that is optimized in a second step. The use of held-out prediction for nuisance components, also known as cross-fitting, is an increasingly popular way of making machine learning methods usable in classical semiparametrics (Schick, 1986; van der Laan & Rose, 2011; Wager et al., 2016; Chernozhukov et al., 2018; Athey & Wager, 2019).

The main difference between this literature and our results is that the existing results typically focus on estimating a single, or low-dimensional, target parameter, whereas we seek to estimate an object $\tau^*(\cdot)$ that may itself be quite complicated. Another research direction that also uses ideas from semiparametrics to estimate complex objects concerns the estimation of optimal treatment allocation rules (Dudík et al., 2011; Zhang et al., 2012; Laber & Zhao, 2015; Luedtke & van der Laan, 2016c; Athey & Wager, 2019). This problem is closely related to, but subtly different from, the problem of estimating $\tau^*(\cdot)$ under squared-error loss; see Kitagawa & Tetenov (2018).

Finally, all the results presented here assume a sampling model, where observations are drawn at random from a population and the target estimand $\tau(\cdot)$ is defined in terms of moments of that population. Ding et al. (2019) considered heterogeneous treatment effect estimation in a strict randomization inference setting, where the features and potential outcomes $\{X_i, Y_i(0), Y_i(1)\}_{i=1}^n$ are taken as fixed and only the treatment W_i is random (Imbens & Rubin, 2015), and showed how to estimate the projection of the realized treatment heterogeneity $Y_i(1) - Y_i(0)$ onto the linear span of the X_i. It would be interesting to explore whether it is possible to derive useful results on nonparametric regularized heterogeneous treatment effect estimation under randomization inference.

4. The R-learner in action

4.1. *Application to a voting study*

To see how the *R*-learner works in practice, we consider an example motivated by Arceneaux et al. (2006), who studied the effect of paid get-out-the-vote calls on voter turnout. A common difficulty in comparing the accuracy of heterogeneous treatment effect estimators on real data is that we do not have access to the ground truth. From this perspective, a major advantage of this example is that Arceneaux et al. (2006) found no effect of get-out-the-vote calls on voter turnout, which suggests that the underlying effect is close to nonexistent. We spike the original dataset with a synthetic treatment effect $\tau^*(\cdot)$ to make the task of estimating heterogeneous treatment effects nontrivial. In other words, both the baseline signal and the propensity scores come from real data, but $\tau^*(\cdot)$ is chosen by us, so we can check whether different methods in fact succeed in recovering it.

The design of Arceneaux et al. (2006) was randomized separately by state and competitiveness of the election, and accounting for varying treatment propensities is necessary for obtaining correct causal effects; a naive analysis ignoring variable treatment propensities estimates the average effect of a single get-out-the-vote call on turnout as 4%, whereas an appropriate analysis finds with high confidence that any treatment effect must be smaller than 1% in absolute value. Although the randomization probabilities were known to the experimenters, here we hide them from our algorithm and require it to learn a model $\hat{e}(\cdot)$ for the treatment propensities. In the original data, not all voters assigned to be contacted could in fact answer the phone call, meaning that all effects should be interpreted as intent-to-treat effects. We focus on $d = 11$ covariates, including state, county, age, gender, etc. Both the outcome Y and the treatment W are binary. The full sample contains 1 895 468 observations, of which 59 264 were assigned treatment. For our analysis, we focused on a subset of 148 160 samples containing all the treated units and a random subset of the controls; thus, two-fifths of our analysis sample was treated. We further divided this sample into a training set of size 100 000, a test set of size 25 000, and a hold-out set with the rest.

As discussed above, for the purpose of this evaluation we assume that the treatment effect in the original data is 0, and we spike in a synthetic treatment effect $\tau^*(X_i) = -\text{VOTE00}_i/(2+100/\text{AGE}_i)$, where VOTE00_i indicates whether the ith unit voted in the year 2000 and AGE_i is that individual's age. Because the outcomes are binary, we add in the synthetic treatment effect by strategically flipping some outcome labels. Denote the original unflipped outcomes by Y_i^*. To add in a treatment effect $\tau^*(\cdot)$, we first draw Bernoulli random variables R_i with probability $|\tau^*(X_i)|$. Then, if $R_i = 0$ we set $Y_i(0) = Y_i(1) = Y_i^*$, whereas if $R_i = 1$ then we set $\{Y_i(0), Y_i(1)\}$ to $(0, 1)$ or $(1, 0)$ depending on whether $\tau^*(X_i) > 0$ or $\tau^*(X_i) < 0$, respectively. Finally, we set $Y_i = Y_i(W_i)$. As is typical in causal inference applications, the treatment heterogeneity here is quite subtle, with $\text{var}\{\tau^*(X)\} = 0.016$, so a large sample size is needed in order to reject a null hypothesis of no treatment heterogeneity.

To use the *R*-learner, we first estimated $\hat{e}(\cdot)$ and $\hat{m}(\cdot)$ to form the *R*-loss function in (4). To do so, we fitted models for the nuisance components via both boosting and the lasso with tuning parameters selected by cross-validation. Then we chose the model that minimized the cross-validated error. This criterion led to our choosing boosting for both $\hat{e}(\cdot)$ and $\hat{m}(\cdot)$. Another option would be to combine the predictions from the lasso and boosting models, as advocated by van der Laan et al. (2007).

Next, we optimized the *R*-loss function. Again, we tried methods based on both the lasso and boosting. This time, the lasso achieved a slightly lower training-set cross-validated *R*-loss than boosting, namely 0.1816 versus 0.1818. Because treatment effects are so weak, and hence there

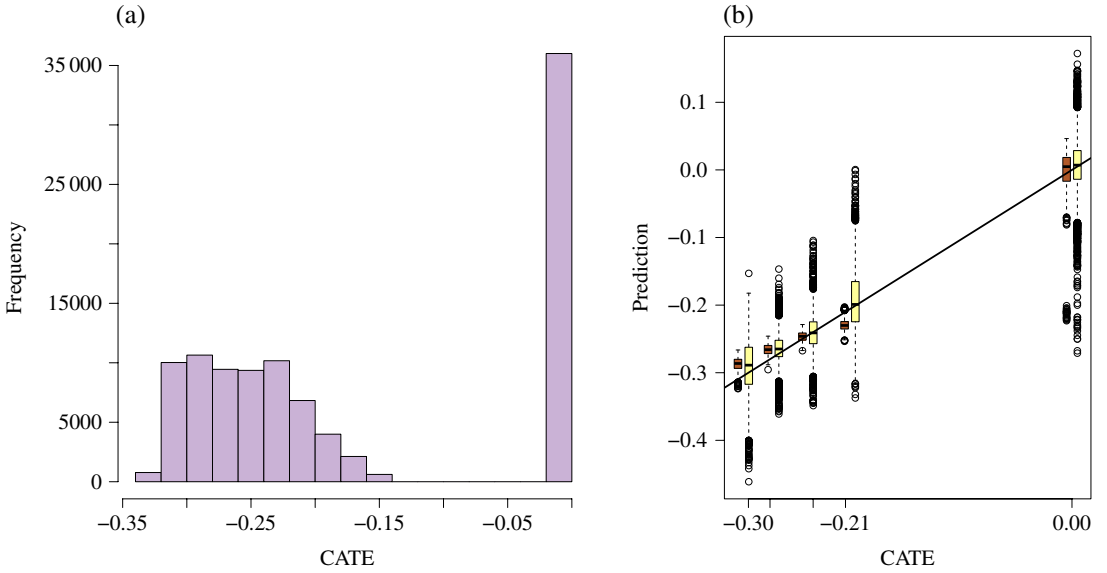

Fig. 1. (a) Histogram showing the distribution of the conditional average treatment effect function $\tau(X_i)$, denoted by CATE, on the test set. (b) Comparison of the true $\tau(X_i)$ (black circles) with estimates $\hat{\tau}(X_i)$ obtained from the R-learner using the lasso (brown boxplots) and boosting (yellow boxplots) to minimize the R-loss, again on the test set; as discussed in § 4.1, both estimates use nuisance components estimated via boosting.

is potential to overfit even in cross-validation, we also examined the R-loss on the hold-out set. The lasso again came out ahead, and the improvement in the R-loss was stable, 0.1781 versus 0.1783. We therefore chose the lasso-based $\hat{\tau}(\cdot)$ fit as our final model for $\tau^*(\cdot)$. Although the improvement in the R-loss is stable, the loss itself is somewhat different between the training and hold-out samples. This appears to be due to the term $n^{-1}\sum_i\{Y_i - \mu^*_{(W_i)}(X_i)\}^2$ induced by irreducible outcome noise. This term is large and noisy in absolute terms; however, it gets cancelled out when comparing the accuracy of two models. This phenomenon plays a key role in understanding the behaviour of model selection via cross-validation (Yang, 2007; Wager, 2020).

Given the constructed conditional average treatment effect function $\tau^*(\cdot)$ in our semi-synthetic data-generative distribution, we can evaluate the oracle test set mean-squared error, $(1/n_{\text{test}})\sum_{\{i\in\text{test}\}}\{\hat{\tau}(X_i) - \tau^*(X_i)\}^2$. Here, it is clear that the lasso did substantially better than boosting, achieving a mean-squared error of 0.47×10^{-3} versus 1.23×10^{-3}. Figure 1(b) compares $\hat{\tau}(\cdot)$ estimates from minimizing the R-loss using the lasso and boosting. The lasso is somewhat biased, but boosting is noisy, and the bias-variance trade-off favours the lasso in this case. With a larger sample size, boosting would be expected to achieve lower mean-squared error.

We also compared our approach with both the single-lasso approach (6) and a popular nonparametric approach to heterogeneous treatment effect estimation via Bayesian additive regression trees (Hill, 2011), with the estimated propensity score added in as a feature following the recommendation of Hahn et al. (2020). The single lasso yielded an oracle test-set error of 0.61×10^{-3}, whereas Bayesian additive regression trees gave an error of 4.05×10^{-3}. Thus it appears that, in this example at least, there is value in using a nonparametric method to estimate $\hat{e}(\cdot)$ and $\hat{m}(\cdot)$, but then using the simpler lasso for $\hat{\tau}(\cdot)$. In contrast, the single-lasso approach uses linear modelling everywhere, leading to potential model misspecification and confounding, whereas the Bayesian additive regression trees approach uses nonparametric modelling everywhere, which can make it difficult to obtain a stable $\tau(\cdot)$ fit. Section 6 has a more comprehensive simulation evaluation of the R-learner relative to several baselines, including the meta-learners of Künzel et al. (2019).

4.2. *Model averaging with the R-learner*

In the previous subsection we looked at an example application, where we were willing to carefully consider the estimation strategies used in each step of the *R*-learner. In other situations, however, a practitioner may prefer to use off-the-shelf treatment effect estimators as the starting point of their analysis. Here we discuss how to use the *R*-learning approach to build a consensus treatment effect estimate via a variant of stacking (Wolpert, 1992; Breiman, 1996; van der Laan et al., 2007; Luedtke & van der Laan, 2016c).

Suppose we start with $k = 1, \ldots, K$ different treatment effect estimators $\hat{\tau}_k$ and we have access to out-of-fold estimates $\hat{\tau}_k^{(-i)}(X_i)$ on our training set. Suppose, moreover, that we have trusted out-of-fold estimates $\hat{e}^{(-i)}(X_i)$ and $\hat{m}^{(-i)}(X_i)$ for the propensity score and main effect, respectively. Then, we propose building a consensus estimate $\hat{\tau}(\cdot)$ by taking the best positive linear combination of the $\hat{\tau}_k(\cdot)$ according to the *R*-loss:

$$\hat{\tau}(x) = \hat{c} + \sum_{k=1}^{K} \alpha_k \hat{\tau}_k(x), \quad (\hat{b}, \hat{c}, \hat{\alpha}) = \underset{b, c, \alpha}{\arg\min} \left\{ \sum_{i=1}^{n} \left[\left\{ Y_i - \hat{m}^{(-i)}(X_i) \right\} - b - \left\{ c + \sum_{k=1}^{K} \alpha_k \hat{\tau}^{(-i)}(X_i) \right\} \left\{ W_i - \hat{e}^{(-i)}(X_i) \right\} \right]^2 : \alpha \geqslant 0 \right\}. \quad (8)$$

For flexibility, we also allow the stacking step (8) to freely adjust a constant treatment effect term c, and we add an intercept b that can be used to absorb any potential bias of $\hat{m}$.

We test this approach on the following data-generation distributions. In both cases, we draw $n = 10\,000$ independent and identically distributed samples from a randomized study design,

$$X_i \sim N(0, I_{d \times d}), \quad W_i \sim \text{Ber}(0.5), \quad Y_i \mid X_i, W_i \sim N\left\{ \frac{3}{1 + \exp(X_{i3} - X_{i2})} + (W_i - 0.5)\, \tau^*(X_i), \sigma^2 \right\} \quad (9)$$

for different choices of $\tau^*(\cdot)$ and σ and with $d = 10$. We consider both a smooth treatment effect function $\tau^*(X_i) = 1/\{1 + \exp(X_{i1} - X_{i2})\}$ and a discontinuous $\tau^*(X_i) = \mathbb{1}\{(X_{i1} > 0)\}/\{1+\exp(-X_{i2})\}$. Given this data-generating process, we tried estimating $\tau(\cdot)$ via Bayesian additive regression trees (Hill, 2011; Hahn et al., 2020), causal forests (Wager & Athey, 2018; Athey et al., 2019), and a stacked combination of the two using (8). We assume that the experimenter knows that the data was randomized, and we used $\hat{e}(x) = 0.5$ anywhere a propensity score was needed. For stacking, we estimated $\hat{m}(\cdot)$ using a random forest.

The results are shown in Fig. 2. In the example with a smooth $\tau^*(\cdot)$, Bayesian additive regression trees slightly outperforms causal forests, while stacking does better than either on its own until the noise level σ gets very large, in which case none of the methods performs much better than a constant treatment effect estimator. Meanwhile, the setting with the discontinuous $\tau^*(\cdot)$ appears to be particularly favourable to causal forests, at least for lower noise levels. Here, stacking is able to automatically match the performance of the more accurate base learner.

5. A QUASI-ORACLE ERROR BOUND

As discussed in the introduction, the high-level goal of our formal analysis is to establish error bounds for *R*-learning that depend only on the complexity of $\tau^*(\cdot)$, and match the error bounds

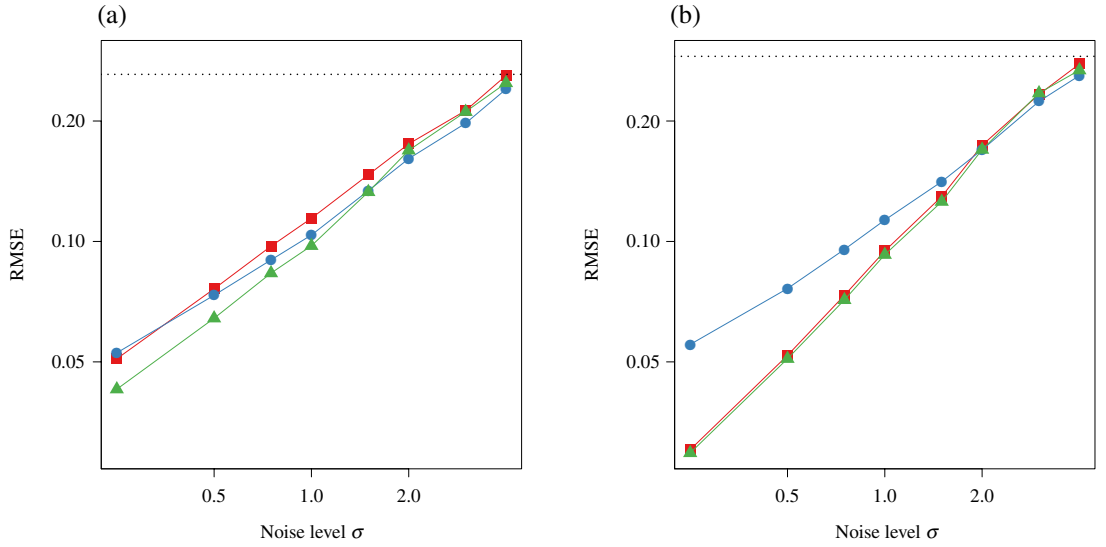

Fig. 2. Root-mean-squared error, RMSE, on the data-generating design (9) with (a) a smooth $\tau^*(\cdot)$, and (b) a discontinuous $\tau^*(\cdot)$, for different noise levels σ. The methods under comparison are the causal forest (red squares), Bayesian additive regression trees (blue circles) and the R-stack (green triangles). For reference, the RMSE of the optimal constant predictor $\tau^*(X_i)$ is shown as a dotted line. All results are aggregated over 50 replications.

that would be achieved if we knew $m^*(\cdot)$ and $e^*(\cdot)$ a priori. To do so, we focus on a variant of the R-learner based on penalized kernel regression. Regularized kernel learning covers a broad class of methods that have been thoroughly studied in the statistical learning literature (see, e.g., Cucker & Smale, 2002; Bartlett & Mendelson, 2006; Caponnetto & De Vito, 2007; Steinwart & Christmann, 2008; Mendelson & Neeman, 2010), and thus provides an ideal case study for examining the asymptotic behaviour of the R-learner.

We study $\|\cdot\|_{\mathcal{H}}$-penalized kernel regression, where $\mathcal{H}$ is a reproducing kernel Hilbert space with a continuous positive-semidefinite kernel function $\mathcal{K}$. Let $\mathcal{P}$ be a nonnegative measure over the compact metric space $\mathcal{X} \subset \mathbb{R}^d$, and let $\mathcal{K}$ be a kernel with respect to $\mathcal{P}$. Let $T_{\mathcal{K}} : L_2(\mathcal{P}) \to L_2(\mathcal{P})$ be defined as $T_{\mathcal{K}}(f)(\cdot) = E\{\mathcal{K}(\cdot, X)f(X)\}$. By Mercer's theorem (Cucker & Smale, 2002), there is an orthonormal basis of eigenfunctions $(\psi_j)_{j=1}^{\infty}$ of $T_{\mathcal{K}}$ with corresponding eigenvalues $(\sigma_j)_{j=1}^{\infty}$ such that $\mathcal{K}(x,y) = \sum_{j=1}^{\infty} \sigma_j \psi_j(x)\psi_j(y)$. Consider the function $\phi : \mathcal{X} \to l_2$ defined by $\phi(x) = \{\sigma_j^{1/2}\psi_j(x)\}_{j=1}^{\infty}$. Following Mendelson & Neeman (2010), we define the reproducing kernel Hilbert space $\mathcal{H}$ to be the image of l_2. For every $t \in l_2$, define the corresponding element in $\mathcal{H}$ by $f_t(x) = \langle \phi(x), t \rangle$ with the induced inner product $\langle f_s, f_t \rangle_{\mathcal{H}} = \langle t, s \rangle$.

Assumption 2. Without loss of generality, $\mathcal{K}(x, x) \leqslant 1$ for all $x \in \mathcal{X}$. For $0 < p < 1$, the eigenvalues σ_j satisfy $G = \sup_{j \geqslant 1} j^{1/p}\sigma_j$ for some constant $G < \infty$, and the orthonormal eigenfunctions $\psi_j(\cdot)$ with $\|\psi_j\|_{L_2(\mathcal{P})} = 1$ are uniformly bounded, i.e., $\sup_j \|\psi_j\|_{\infty} \leqslant A < \infty$. Finally, the outcomes Y_i are almost surely bounded, i.e., $|Y_i| \leqslant M$.

Assumption 3. The true conditional average treatment effect function $\tau^*(x) = E\{Y_i(1) - Y_i(0) \mid X_i = x\}$ satisfies $\|T_{\mathcal{K}}^{\alpha}\{\tau^*(\cdot)\}\|_{\mathcal{H}} < \infty$ for some $0 < \alpha < 1/2$.

To interpret the assumption above, we do not assume that $\tau^*(\cdot)$ has a finite $\mathcal{H}$-norm; rather, we only assume that we can make it have a finite $\mathcal{H}$-norm after a sufficient amount of smoothing. More concretely, with $\alpha = 0$, $T_{\mathcal{K}}^{\alpha}$ would be the identity operator, and so this assumption would

be equivalent to the strongest possible assumption that $\|\tau^*(\cdot)\|_{\mathcal{H}} < \infty$ itself. Then, as α grows, the assumption gets progressively weaker, and at $\alpha = 1/2$ it would reduce to simply requiring that $\tau^*(\cdot)$ belong to the space $L_2(\mathcal{P})$ of square-integrable functions.

We study oracle penalized regressions $\tilde{\tau}(\cdot)$ that minimize the objective

$$\tilde{\tau}(\cdot) = \arg\min \left(\frac{1}{n} \sum_{i=1}^{n} \left[\{Y_i - m^*(X_i)\} - \{W_i - e^*(X_i)\} \tau(X_i) \right]^2 + \Lambda_n(\|\tau\|_{\mathcal{H}}) : \|\tau\|_\infty \leqslant 2M \right), \tag{10}$$

as well as feasible analogues obtained by cross-fitting (Schick, 1986; Chernozhukov et al., 2018),

$$\hat{\tau}(\cdot) = \arg\min_{\tau \in \mathcal{H}} \left(\frac{1}{n} \sum_{i=1}^{n} \left[\{Y_i - \hat{m}^{(-q(i))}(X_i)\} - \{W_i - \hat{e}^{(-q(i))}(X_i)\} \tau(X_i) \right]^2 + \Lambda_n(\|\tau\|_{\mathcal{H}}) : \|\tau\|_\infty \leqslant 2M \right). \tag{11}$$

Adding the upper bound $\|\tau\|_\infty \leqslant 2M$, or in fact any finite upper bound on τ, enables us to rule out some pathological behaviours.

We seek to characterize the accuracy of our estimator $\hat{\tau}(\cdot)$ by bounding its regret $R(\hat{\tau})$,

$$R(\tau) = L(\tau) - L(\tau^*), \quad L(\tau) = E\left(\left[\{Y_i - m^*(X_i)\} - \tau(X_i)\{W_i - e^*(X_i)\} \right]^2 \right).$$

Recall that, by the expansion (1), we have $E\{Y_i - m^*(X_i) \mid X_i, W_i\} = \tau^*(X_i)\{W_i - e^*(X_i)\}$, implying that

$$L(\tau) = E\left[\text{var}\{Y_i - m^*(X_i) \mid X_i, W_i\} \right] + E\left[\{\tau(X_i) - \tau^*(X_i)\}^2 \{W_i - e^*(X_i)\}^2 \right]$$

and $R(\tau) = E[\{\tau(X_i) - \tau^*(X_i)\}^2 \{W_i - e^*(X_i)\}^2]$. Thus, if we have overlap, i.e., if there is an $\eta > 0$ such that $\eta < e^*(x) < 1 - \eta$ for all $x \in \mathcal{X}$, then

$$(1-\eta)^{-2} R(\tau) < E\left[\{\tau(X_i) - \tau^*(X_i)\}^2 \right] < \eta^{-2} R(\tau), \tag{12}$$

meaning that regret bounds translate into squared-error loss bounds for $\tau(\cdot)$ and vice versa. We note that when the overlap parameter η gets close to 0, the coupling (12) becomes fairly loose.

The sharpest regret bounds for the oracle learner (10) under Assumptions 2 and 3 are due to Mendelson & Neeman (2010), see also Steinwart et al. (2009) and scale as

$$R(\tilde{\tau}) = \tilde{O}_{\text{P}}\left\{ n^{-\frac{1-2\alpha}{p+(1-2\alpha)}} \right\}, \tag{13}$$

where the $\tilde{O}_{\text{P}}$ notation hides logarithmic factors. In the case of $\alpha = 0$, where τ^* is within the reproducing kernel Hilbert space used for penalization, we recover the more familiar $n^{-1/(1+p)}$ rate established by Caponnetto & De Vito (2007). Again, our goal is to establish excess loss bounds for our feasible estimator $\hat{\tau}$ that match the bound (13) available to the oracle that knows $m^*(\cdot)$ and $e^*(\cdot)$ a priori.

In order to do so, we first need to briefly review the proof techniques underlying (13). The argument of Mendelson & Neeman (2010) relies on a quasi-isomorphic coordinate projection lemma of Bartlett (2008). To state this result, write

$$\mathcal{H}_c = \{\tau : \|\tau\|_{\mathcal{H}} \leqslant c,\ \|\tau\|_\infty \leqslant 2M\} \tag{14}$$

for the radius-c ball of $\mathcal{H}$ capped by $2M$, let $\tau_c^* = \arg\min\{L(\tau) : \tau \in \mathcal{H}_c\}$ denote the best approximation to τ^* within $\mathcal{H}_c$, and define the c-regret $R(\tau;\, c) = L(\tau) - L(\tau_c^*)$ over $\tau \in \mathcal{H}_c$. We also define the estimated and oracle c-regret functions $\hat{R}_n$ and $\tilde{R}_n$ in terms of the estimated and oracle losses $\hat{L}_n$ and $\tilde{L}_n$:

$$\hat{R}_n(\tau;\, c) = \hat{L}_n(\tau) - \hat{L}_n(\tau_c^*), \quad \tilde{R}_n(\tau;\, c) = \tilde{L}_n(\tau) - \tilde{L}_n(\tau_c^*),$$

$$\tilde{L}_n(\tau) = \frac{1}{n}\sum_{i=1}^n \left[Y_i - m^*(X_i) - \tau(X_i)\{W_i - e^*(X_i)\}\right]^2,$$

$$\hat{L}_n(\tau) = \frac{1}{n}\sum_{i=1}^n \left[Y_i - \hat{m}^{(-q(i))}(X_i) - \tau(X_i)\{W_i - \hat{e}^{(-q(i))}(X_i)\}\right]^2.$$

The function $\hat{R}_n(\tau;\, c)$ is not observable as it depends on τ_c^*; however, this does not hinder us from establishing high-probability bounds for it. The lemma below is adapted from Bartlett (2008).

LEMMA 1. *Let $\check{L}_n(\tau)$ be any loss function, and let $\check{R}_n(\tau;\, c) = \check{L}_n(\tau) - \check{L}_n(\tau_c^*)$ be the associated regret. Let $\rho_n(c)$ be a continuous positive function that is increasing in c. Suppose that for every $1 \leqslant c \leqslant C$ and some $k > 1$, the following inequality holds:*

$$\frac{1}{k}\check{R}_n(\tau;\, c) - \rho_n(c) \leqslant R(\tau;\, c) \leqslant k\check{R}_n(\tau;\, c) + \rho_n(c), \quad \tau \in \mathcal{H}_c. \tag{15}$$

Then, writing $\kappa_1 = 2k + 1/k$ and $\kappa_2 = 2k^2 + 3$, any solution to the empirical minimization problem with regularizer $\Lambda_n(c) \geqslant \rho_n(c)$,

$$\check{\tau} \in \arg\min_{\tau \in \mathcal{H}_C}\{\check{L}(\tau) + \kappa_1 \Lambda_n(\|\tau\|_{\mathcal{H}})\},$$

also satisfies the following risk bound:

$$L(\check{\tau}) \leqslant \inf_{\tau \in \mathcal{H}_C}\left\{L(\tau) + \kappa_2 \Lambda_n(\|\tau\|_{\mathcal{H}})\right\}.$$

In other words, the above lemma reduces the problem of deriving regret bounds to establishing quasi-isomorphisms as in (15), and any with-high-probability quasi-isomorphism guarantee yields a with-high-probability regret bound. In particular, we can use this approach to prove the regret bound (13) for the oracle learner as follows. First we need a with-high-probability quasi-isomorphism of the form

$$\frac{1}{k}\tilde{R}_n(\tau;\, c) - \rho_n(c) \leqslant R(\tau;\, c) \leqslant k\tilde{R}_n(\tau;\, c) + \rho_n(c). \tag{16}$$

Mendelson & Neeman (2010) provide such a bound for $\rho_n(c)$ that scales as

$$\rho_n(c) \sim \{1 + \log(n) + \log\log(c + e)\}\left\{\frac{(c+1)^p \log(n)}{\sqrt{n}}\right\}^{2/(1+p)}. \tag{17}$$

Lemma 1 then immediately implies that penalized regression over $\mathcal{H}_C$ with the oracle loss function $\tilde{L}(\cdot)$ and regularizer $\kappa_1 \rho_n(c)$ satisfies the following bound with high probability:

$$R(\tilde{\tau}) = L(\tilde{\tau}) - L\big(\tau^*\big) \leqslant \inf_{\tau \in \mathcal{H}_C} \{L(\tau) + \kappa_2 \rho_n(\|\tau\|_{\mathcal{H}})\} - L\big(\tau^*\big).$$

Furthermore, following Mendelson & Neeman (2010, Corollary 2.7), for any $1 \leqslant c \leqslant C$ we also have

$$\inf_{\tau \in \mathcal{H}_C} \{L(\tau) + \kappa_2 \rho_n(\|\tau\|_{\mathcal{H}})\} \leqslant L\big(\tau^*\big) + \big\{L\big(\tau_c^*\big) - L\big(\tau^*\big)\big\} + \kappa_2 \rho_n(c). \tag{18}$$

Mendelson & Neeman (2010) considered the case where $C = \infty$; here, we only take C large enough for our argument; see the proof for details. Finally, with the scaling of $\rho_n(c)$ in (17) and the approximation error bound

$$L\big(\tau_c^*\big) - L\big(\tau^*\big) \leqslant c^{(2\alpha-1)/\alpha} \left\| T_{\mathcal{K}}^{\alpha} \{\tau^*(\cdot)\} \right\|_{\mathcal{H}}^{1/\alpha} \tag{19}$$

established by Smale & Zhou (2003) in the setting of Assumption 3, we obtain a practical regret bound by choosing $c = c_n$ to optimize the right-hand side of (18). The specific rate in (13) comes from setting $c_n = n^{\alpha/\{p+(1-2\alpha)\}}$.

The outcome is that if we can match the strength of the quasi-isomorphism bounds (16) with our feasible loss function, i.e., obtain an analogous bound in terms of $\hat{R}_n$ as opposed to $\tilde{R}_n$, then we can also match the rate of any regret bounds proved using the above argument. The proof of the following result relies on several concentration results, including Talagrand's inequality and generic chaining (Talagrand, 2006), and makes heavy use of cross-fitting-style arguments (Schick, 1986; van der Laan & Rose, 2011; Chernozhukov et al., 2018).

LEMMA 2. *Under the conditions of Lemma 1, suppose that the propensity estimate $\hat{e}(x)$ is uniformly consistent, $\xi_n := \sup_{x \in \mathcal{X}} \left|\hat{e}(x) - e^*(x)\right| \to 0$ in probability, and the L_2 errors converge at the rate*

$$E\big[\{\hat{m}(X) - m^*(X)\}^2\big],\ E\big[\{\hat{e}(X) - e^*(X)\}^2\big] = O(a_n^2)$$

for some sequence a_n such that $a_n = O(n^{-\kappa})$ with $\kappa > 1/4$. Suppose, moreover, that we have overlap, i.e., $\eta < e^*(x) < 1 - \eta$ for some $\eta > 0$, and that Assumptions 2 and 3 hold. Then

$$\left|\hat{R}_n(\tau;\, c) - \tilde{R}_n(\tau;\, c)\right| \leqslant 0.125\, R(\tau;\, c) + o\{\rho_n(c)\} \tag{20}$$

with probability at least $1 - \varepsilon$, for all $\tau \in \mathcal{H}_c$ and $1 \leqslant c \leqslant c_n \log(n)$ with $c_n = n^{\alpha/(p+1-2\alpha)}$ for large enough n.

This result implies that we can turn any quasi-isomorphism (16) for the oracle learner with error $\rho_n(c)$ into a quasi-isomoprhism bound for $\hat{R}(\tau)$ with error inflated by the right-hand side of (20). Thus, given any regret bound for the oracle learner constructed using Lemma 1, we can also get an analogous regret bound for the feasible learner provided we regularize just a little bit more. The following result makes this formal.

THEOREM 1. *Given the conditions of Lemma 2 and that $2\alpha < 1 - p$, suppose that we obtain $\hat{\tau}(\cdot)$ via a penalized kernel regression variant of the R-learner (11), with a properly chosen penalty*

of the form $\Lambda_n(\|\hat{\tau}\|_{\mathcal{H}})$ *specified in the proof. Then* $\hat{\tau}(\cdot)$ *satisfies the same regret bound* (13) *as* $\tilde{\tau}(\cdot)$*, i.e.,* $R(\hat{\tau}) = O_{\mathrm{P}}\left[n^{-(1-2\alpha)/\{p+(1-2\alpha)\}}\right]$.

In other words, we have found that with penalized kernel regression, the *R*-learner can match the best available performance guarantees available for the oracle learner (10) that knows everything about the data-generating distribution except the true treatment effect function, and both the feasible learner and the oracle learner satisfy

$$R(\hat{\tau}),\, R(\tilde{\tau}) = \tilde{O}_{\mathrm{P}}(r_n^2), \quad r_n = n^{-(1-2\alpha)/[2\{p+(1-2\alpha)\}]}. \tag{21}$$

As we approach the semiparametric case, i.e., as $\alpha, p \to 0$, we recover the well-known result from the semiparametric inference literature that, to get $n^{-1/2}$-consistent inference for a single target parameter, we need fourth-root-consistent nuisance parameter estimates; see Chernozhukov et al. (2018) for a review and references. After a first draft of the present paper was disseminated, several authors established further quasi-oracle-type results for the *R*-learner and related methods; see, in particular, Foster & Syrgkanis (2019) and Kennedy (2020).

We emphasize that our quasi-oracle result depends on a local robustness property of the *R*-loss function and does not hold for general meta-learners; for example, it does not hold for the *X*-learner of Künzel et al. (2019). To see this, we argue by contradiction. We show that it is possible to make $o(n^{-1/4})$ changes to the nuisance components $\hat{\mu}_{(w)}(x)$ used by the *X*-learner which induce changes in the *X*-learner's $\hat{\tau}(\cdot)$ estimates that dominate the error scale in (21). Thus, there must be some choices of $o(n^{-1/4})$-consistent $\hat{\mu}_{(w)}(x)$ with which the *X*-learner will not converge at the rate (21). The contradiction arises as follows. Pick $\xi > 0$ such that $0.25 + \xi < (1-2\alpha)/[2\{p+(1-2\alpha)\}]$, and modify the nuisance components used to form the *X*-learner in (7) such that $\hat{\mu}_{(0)}(x) \leftarrow \hat{\mu}_{(0)}(x) - c/n^{0.25+\xi}$ and $\hat{\mu}_{(1)}(x) \leftarrow \hat{\mu}_{(1)}(x) + c/n^{0.25+\xi}$. Recall that the *X*-learner fits $\hat{\tau}_{(1)}(\cdot)$ by minimizing $n_1^{-1}\sum_{W_i=1}\{Y_i - \hat{\mu}_{(0)}^{(-i)}(X_i) - \tau_{(1)}(X_i)\}^2$ and fits $\hat{\tau}_{(0)}(\cdot)$ by solving an analogous problem on the controlled units. Combining the $\hat{\tau}_{(w)}$ estimates from these two loss functions, we observe that the final estimate of the treatment effect is also shifted by $\hat{\tau}(x) \leftarrow \hat{\tau}(x) + c/n^{0.25+\xi}$. The perturbations $c/n^{0.25+\xi}$ are vanishingly small on the $n^{-1/4}$ scale, and so would not affect conditions analogous to those of Theorem 1; yet they have a big enough effect on $\hat{\tau}(x)$ to break any convergence results on the scale of (21). Künzel et al. (2019) did obtain some quasi-oracle-type results; however, they focused only on the case where the number of control units $|\{W_i = 0\}|$ grows much faster than the number of treated units $|\{W_i = 1\}|$. They showed that in this case the *X*-learner performs as well as an oracle that already knew the mean response function for the controls, $\mu^*_{(0)}(x) = E(Y_i(0) \mid X_i = x)$. Intriguingly, in this special case, we have $m^*(x) \approx \mu^*_{(0)}(x)$ and $e^*(x) \approx 0$, and so the *R*-learner in (11) is roughly equivalent to the *X*-learner procedure (7). Thus, at least qualitatively, we can interpret the result of Künzel et al. (2019) as a special case of our result in the case where the number of controls dominates the number of treated units, or vice versa.

6. SIMULATION EXPERIMENTS

6.1. *Baseline methods and simulation set-ups*

Our approach to heterogeneous treatment effect estimation via learning objectives can be implemented using any method that is framed as a loss-minimization problem, such as boosting and decision trees. In this section, we focus on simulation experiments using the *R*-learner, a direct implementation of (4) based on the lasso, kernel ridge regression and boosting. We

follow the terminology of Künzel et al. (2019) and use the following methods for heterogeneous treatment effect estimation as baselines. The S-learner fits a single model for $f(x, w) = E(Y \mid X = x, W = w)$ and then estimates $\hat{\tau}(x) = \hat{f}(x, 1) - \hat{f}(x, 0)$; the T-learner fits the functions $\mu^*_{(w)}(x) = E(Y \mid X = x, W = w)$ separately for $w \in \{0, 1\}$ and then estimates $\hat{\tau}(x) = \hat{\mu}_{(1)}(x) - \hat{\mu}_{(0)}(x)$; the X-learner and U-learner are as described in § 3. In addition, for the boosting-based experiments, we consider the causal boosting algorithm proposed by Powers et al. (2018), denoted by CB in § 6.4.

Finally, for the lasso-based experiments, we consider an additional variant of our method, the *RS*-learner, which in spirit combines the R- and S-learners by including an additional term in the loss function, and then separately penalizes the main and treatment effect terms as in Imai & Ratkovic (2013). Specifically, we use $\hat{\tau}(x) = x^{\mathrm{T}}\hat{\delta}$, where $\hat{b}$ and $\hat{\delta}$ minimize

$$\frac{1}{n}\sum_{i=1}^{n}\left[Y_i - \hat{m}^{(-i)}(X_i) - X_i^{\mathrm{T}}b - \{W_i - \hat{e}^{(-i)}(X_i)\}X_i^{\mathrm{T}}\delta\right]^2 + \lambda(\|b\|_1 + \|\delta\|_1).$$

Heuristically, one hopes that the *RS*-learner would be more robust, as it has an additional term to eliminate confounders.

In all simulations, we generate data as follows. For different choices of the X-distribution P_d indexed by dimension d, noise level σ, propensity function $e^*(\cdot)$, baseline main effect $b^*(\cdot)$ and treatment effect function $\tau^*(\cdot)$, we generate

$$X_i \sim P_d, \quad W_i \mid X_i \sim \mathrm{Ber}\{e^*(X_i)\}, \quad \varepsilon_i \mid X_i \sim N(0, 1),$$
$$Y_i = b^*(X_i) + (W_i - 0.5)\tau^*(X_i) + \sigma\varepsilon_i.$$

We consider the following specific simulation designs. Set-up A has difficult nuisance components and an easy treatment effect function. We use the scaled Friedman (1991) function for the baseline main effect $b^*(X_i) = \sin(\pi X_{i1}X_{i2}) + 2(X_{i3} - 0.5)^2 + X_{i4} + 0.5X_{i5}$, along with $X_i \sim \mathrm{Un}(0, 1)^d$, $e^*(X_i) = \mathrm{trim}_{0.1}\{\sin(\pi X_{i1}X_{i2})\}$ and $\tau^*(X_i) = (X_{i1} + X_{i2})/2$, where $\mathrm{trim}_\eta(x) = \max\{\eta, \min(x, 1 - \eta)\}$. Set-up B employs a randomized trial. Here, $e^*(x) = 1/2$ for all $x \in \mathbb{R}^d$, so it is possible to be accurate without explicitly controlling for confounding. We take $X_i \sim N(0, I_{d\times d})$, $\tau^*(X_i) = X_{i1} + \log\{1 + \exp(X_{i2})\}$ and $b^*(X_i) = \max\{X_{i1} + X_{i2}, X_{i3}, 0\} + \max\{X_{i4} + X_{i5}, 0\}$. Set-up C has an easy propensity score and a difficult baseline. In this set-up there is strong confounding, but the propensity score is much easier to estimate than the baseline: $X_i \sim N(0, I_{d\times d})$, $e^*(X_i) = 1/\{1 + \exp(X_{i2} + X_{i3})\}$, $b^*(X_i) = 2\log\{1 + \exp(X_{i1} + X_{i2} + X_{i3})\}$, and the treatment effect is constant, $\tau^*(X_i) = 1$. Set-up D has unrelated treatment and control arms, with data generated as $X_i \sim N(0, I_{d\times d})$, $e^*(X_i) = 1/\{1+\exp(-X_{i1})+\exp(-X_{i2})\}$, $\tau^*(X_i) = \max\{X_{i1} + X_{i2} + X_{i3}, 0\} - \max\{X_{i4} + X_{i5}, 0\}$ and $b^*(X_i) = [\max\{X_{i1} + X_{i2} + X_{i3}, 0\} + \max\{X_{i4} + X_{i5}, 0\}]/2$. Here, $\mu^*_{(0)}(X)$ and $\mu^*_{(1)}(X)$ are uncorrelated, and so there is no benefit to learning them jointly.

6.2. *Lasso-based experiments*

In this section, we compare S-, T-, X-, U-, and our R- and *RS*-learners implemented via the lasso on simulated designs. For the S-learner we follow Imai & Ratkovic (2013) in using (6), while for the T-learner we use (5). For the X-, R- and *RS*-learners, we use L_1-penalized logistic regression to estimate the propensity $\hat{e}$ and use the lasso for all other regression estimates.

For all estimators, we run the lasso on the pairwise interactions of a natural spline basis expansion with seven degrees of freedom on X_i. We generate n data points as the training set

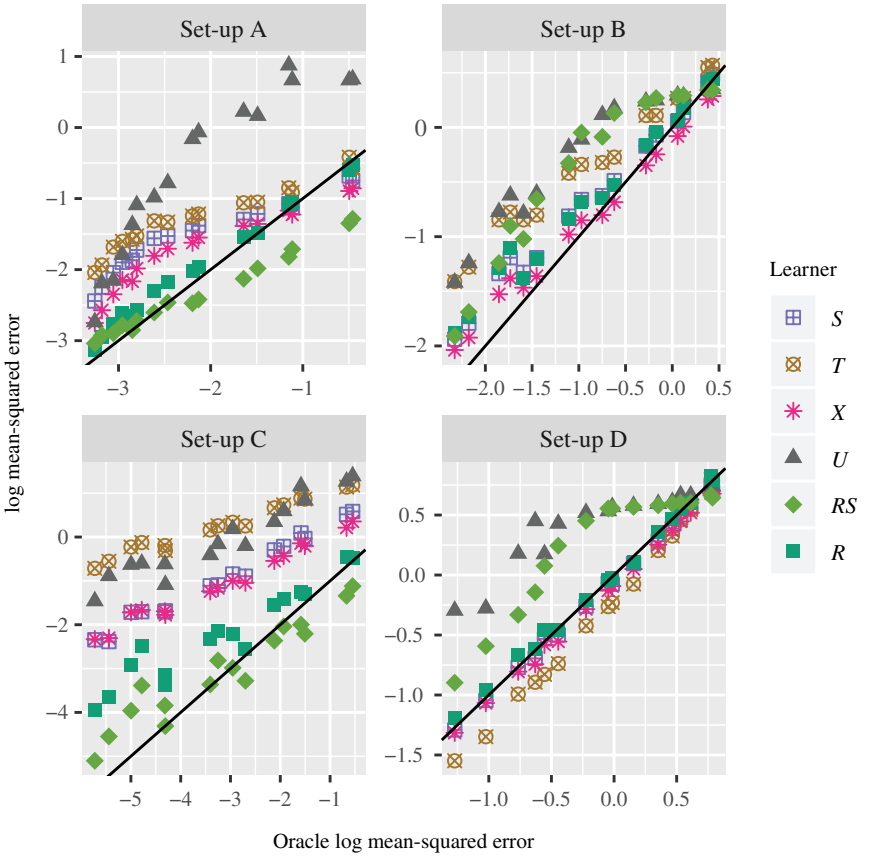

Fig. 3. Performance of lasso-based S-, T-, X-, U-, RS- and R-learners, relative to a lasso-based oracle learner (3), in simulation set-ups A–D described in § 6: set-up A has complicated nuisance components, but a simple $\tau(\cdot)$ function; set-up B is a randomized trial; set-up C has a simple propensity function, but a complicated main effect function; and set-up D has unrelated treatment and control response surfaces. Plotted are results for all combinations of $n \in \{500, 1000\}$, $d \in \{6, 12\}$ and $\sigma \in \{0.5, 1, 2, 3\}$, and each point in the plots represents the average performance of one learner for one of these 16 parameter specifications. All mean-squared error values are aggregated over 500 runs and reported on an independent test set, and are plotted on the logarithmic scale.

and generate a separate test set also with n data points; the reported mean-squared error is on the test set. The penalty parameter is chosen by 10-fold cross-validation. For the R- and RS-learners, we use 10-fold cross-fitting on $\hat{e}$ and $\hat{m}$ in (4). All methods are implemented with `glmnet` (Friedman et al., 2010). The U-learner suffers from high variance and instability due to dividing by the propensity estimates. Therefore, we set a cut-off for the propensity estimate at level 0.05. We have also found empirically that the U-learner gives much lower estimation error if we choose to use the largest regularization parameter that achieves one standard error away from the minimum in the cross-validation step. Therefore the U-learner uses `lambda.1se` as its cross-validation parameter; the other learners use `lambda.min` in `glmnet`.

In Fig. 3, we compare the performance of the six methods under consideration with an oracle that runs the lasso on (3), for different values of the sample size n, dimension d and noise level σ. As is clear from these results, the simulation settings differ vastly in difficulty, both in terms of the accuracy of the oracle and in terms of the ability of feasible methods to approach the oracle. The raw numbers plotted in Fig. 3 are available in the Supplementary Material.

In set-ups A and C, where there is complicated confounding that needs to be overcome before we can estimate a simple treatment effect function $\tau^*(\cdot)$, the R- and RS-learners stand out. All the

methods do reasonably well in the randomized trial, namely set-up B, where it was not necessary to adjust for confounding, with the X-, S- and R-learners performing best. Finally, having completely disjoint functions for the treated and control arms is unusual in practice; nevertheless, we consider this possibility in set-up D, where there is no reason to model $\mu^*_{(0)}(x)$ and $\mu^*_{(1)}(x)$ jointly, and find that the T-learner, which in fact models them separately, performs well.

Overall, the R- and RS-learners consistently achieve good performance and, in most simulation settings, essentially match the performance of the oracle (3) in terms of mean-squared error. The U-learner suffers from high loss due to its instability.

6.3. *Kernel ridge regression-based experiments*

We move on to comparing S-, T-, X-, U- and R-learners implemented via kernel ridge regression with a Gaussian kernel. We use a variant of the `KRLS` package (Ferwerda et al., 2017; R Development Core Team, 2021), available at `https://github.com/lukesonnet/KRLS`, that allows weighted regression. For fitting the objective in each subroutine in all methods, we run a five-fold cross-validation to search through the width of the Gaussian kernel and the ridge regularization parameter both from a grid of 10^k with $k \in \{-3, -2.5, -2, \ldots, 2, 2.5, 3\}$. We experiment on the same set-ups and parameter variations, including variations on the sample size n, dimension d and noise level σ, as in § 6.2. The results are depicted in Fig. 4, and the raw values are given in the Supplementary Material. From Fig. 4 we again observe that the R-learner does particularly well in set-ups A and C, where the treatment effect functions are relatively simple and the treatment propensity is not constant.

6.4. *Gradient boosting-based experiments*

Finally, we compare S-, T-, X-, U- and R-learners implemented via gradient boosting, as well as the causal boosting algorithm, denoted by CB. We use the `R` package `causalLearning` (R Development Core Team, 2021) for CB, while all the other methods are implemented using `XGboost` (Chen & Guestrin, 2016). To fit the objective in each subroutine in all methods, we draw a random set of 10 combinations of hyperparameters from the following grid: `subsample` = [0.5, 0.75, 1], `colsample_bytree` = [0.6, 0.8, 1], `eta` = [5e-3, 1e-2, 1.5e-2, 2.5e-2, 5e-2, 8e-2, 1e-1, 2e-1], `max_depth` = $[3, \ldots, 20]$, `gamma` = `Uniform`(0, 0.2), `min_child_weight` = $[1, \ldots, 20]$, `max_delta_step` = $[1, \ldots, 10]$. Then we cross-validate on the number of boosted trees for each combination with an early stopping of 10 iterations. We experiment on the same set-ups and parameter variations, including variations on the sample size n, dimension d and noise level σ, as in § 6.2. The results are depicted in Fig. 5, and the raw values are given in the Supplementary Material.

In Fig. 5, the R-learner again stands out in set-ups A and C. All methods perform reasonably well in the randomized control setting of set-up B. In set-up D, the T-learner performs best since the treated and control arms are generated from unrelated functions.

In both sets of experiments, for simplicity of illustration we have used the lasso, kernel ridge regression and boosting to learn $\hat{m}(\cdot)$ and $\hat{e}(\cdot)$. In practice, we recommend cross-validating on a variety of black-box learners, such as the lasso, random forests and neural networks, that are tuned for prediction accuracy to learn these two pilot quantities. All simulation results above can be replicated using the publicly available `rlearner` package for `R`, available at `https://github.com/xnie/rlearner`.

7. Discussion and extensions

A natural generalization of our framework arises when, in some applications, one needs to deal with multiple treatment options. For example, in medicine, we may want to compare a control

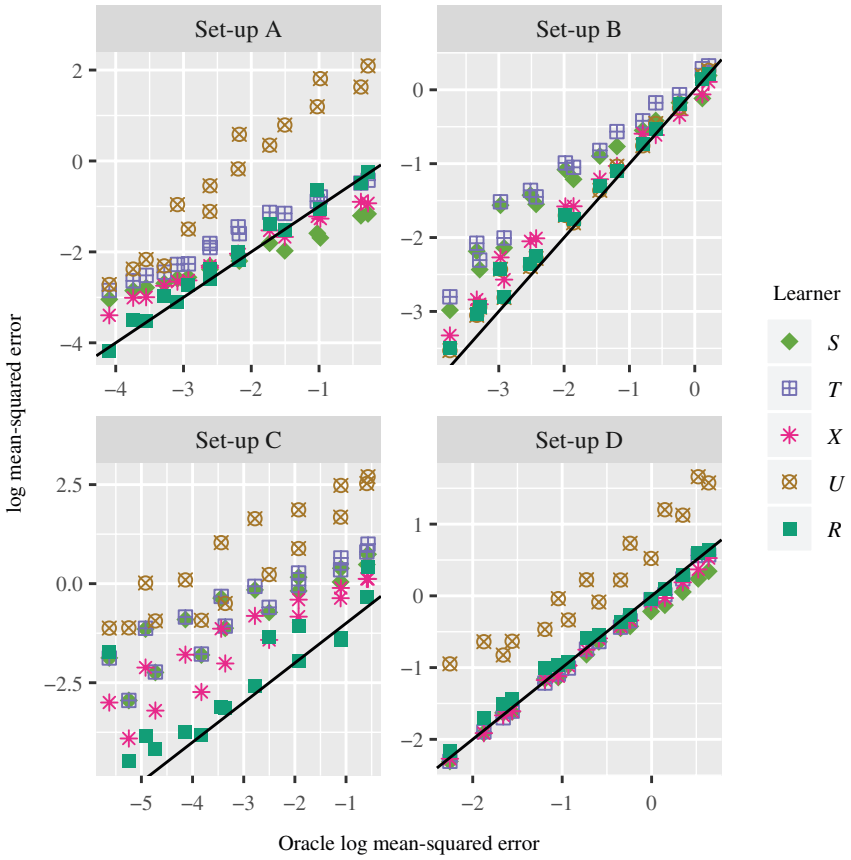

Fig. 4. Performance of S-, T-, X-, U-, RS- and R-learners, relative to an oracle learner (3), all based on kernel ridge regression with a Gaussian kernel, in simulation set-ups A–D described in § 6. Plotted are results for all combinations of $n \in \{500, 1000\}$, $d \in \{6, 12\}$ and $\sigma \in \{0.5, 1, 2, 3\}$, and each point in the plots represents the average performance of one learner for one of these 16 parameter specifications. All mean-squared error values are aggregated over 200 runs and reported on an independent test set, and are plotted on the logarithmic scale.

condition with multiple different experimental treatments. If there are k different treatments along with a control arm, we can encode $W \in \{0, 1\}^k$, and a multivariate version of Robinson's transformation suggests the estimator

$$\hat{\tau}(\cdot) = \arg\min_{\tau} \left(\frac{1}{n} \sum_{i=1}^{n} \left[\left\{ Y_i - \hat{m}^{(-i)}(X_i) \right\} - \left\langle W_i - \hat{e}^{(-i)}(X_i), \tau(X_i) \right\rangle \right]^2 + \Lambda_n\{\tau(\cdot)\} \right),$$

where the angle brackets indicate an inner product, $e(x) = E(W \mid X = x) \in \mathbb{R}^k$ is a vector, and $\tau_l(x)$ measures the conditional average treatment effect of the lth treatment arm at $X_i = x$, for $l = 1, \ldots, k$. When implementing variants of this approach in practice, different choices of $\Lambda_n\{\tau(\cdot)\}$ may be needed to reflect relationships between the treatment effects of different arms, such as whether there is a natural ordering of treatment arms or if there are some arms that are believed a priori to have similar effects.

It would also be interesting to consider extensions of the R-learner to cases where the treatment assignment W_i is not unconfounded, and one needs to rely on an instrument to identify causal effects. Chernozhukov et al. (2018) discusses how Robinson's approach to the partially linear model generalizes naturally to this case, and Athey et al. (2019) have adapted their causal forest

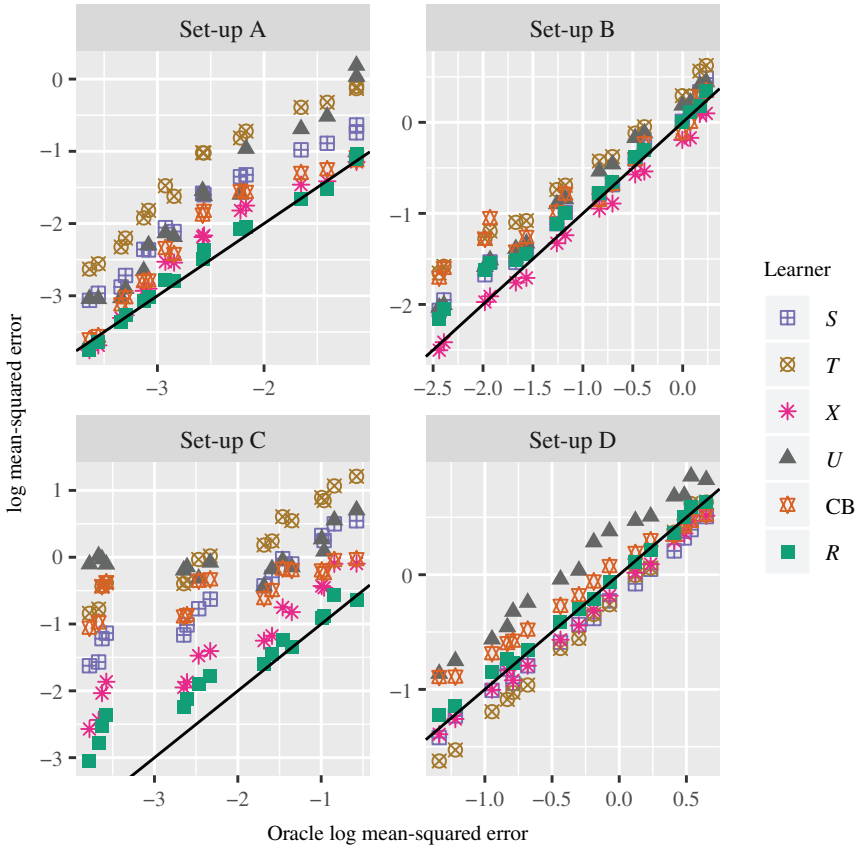

Fig. 5. Performance of boosting-based S-, T-, X-, U- and R-learners as well as causal boosting, CB, relative to a boosting-based oracle learner (3), in simulation set-ups A–D described in § 6. Plotted are results for all combinations of $n \in \{500, 1000\}$, $d \in \{6, 12\}$ and $\sigma \in \{0.5, 1, 2, 3\}$, and each point on the plots represents the average performance of one learner for one of these 16 parameter specifications. All mean-squared error numbers are aggregated over 200 runs and reported on an independent test set, and are plotted on the logarithmic scale.

to work with instruments. The underlying estimating equations, however, cannot be interpreted as loss functions as easily as (3), especially in the case where instruments may be weak, and so we leave this extension of the R-learner to future work.

Acknowledgement

We are grateful for enlightening conversations with Susan Athey, Emma Brunskill, John Duchi, Tatsunori Hashimoto, Guido Imbens, Sören Künzel, Percy Liang, Whitney Newey, Mark van der Laan, Alejandro Schuler, Robert Tibshirani, Bin Yu and Yuchen Zhang, as well as for helpful comments and feedback from seminar participants at several universities and workshops, and from the referees. This research was partially supported by the U.S. National Science Foundation, a Facebook Faculty Award, and a 2018 Stanford Human-Centered AI seed grant. Nie was awarded a Thomas R. Ten Have Award based on this work at the 2018 Atlantic Causal Inference Conference.

Supplementary material

Supplementary material available at *Biometrika* online includes detailed simulation results and proofs of Lemmas 1 and 2 and Theorem 1.

REFERENCES

ABADI, M., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M. et al. (2016). TensorFlow: A system for large-scale machine learning. In *Proc. 12th USENIX Sympos. Operating Systems Design and Implementation (OSDI'16)*. Berkeley, California: USENIX Association, pp. 265–83.

ARCENEAUX, K., GERBER, A. S. & GREEN, D. P. (2006). Comparing experimental and matching methods using a large-scale voter mobilization experiment. *Polit. Anal.* **14**, 37–62.

ATHEY, S. (2017). Beyond prediction: Using big data for policy problems. *Science* **355**, 483–5.

ATHEY, S. & IMBENS, G. (2016). Recursive partitioning for heterogeneous causal effects. *Proc. Nat. Acad. Sci.* **113**, 7353–60.

ATHEY, S., TIBSHIRANI, J. & WAGER, S. (2019). Generalized random forests. *Ann. Statist.* **47**, 1148–78.

ATHEY, S. & WAGER, S. (2019). Efficient policy learning. *arXiv:* 1702.02896v5.

BARTLETT, P. L. (2008). Fast rates for estimation error and oracle inequalities for model selection. *Economet. Theory* **24**, 545–52.

BARTLETT, P. L. & MENDELSON, S. (2006). Empirical minimization. *Prob. Theory Rel. Fields* **135**, 311–34.

BELLONI, A., CHERNOZHUKOV, V., FERNÁNDEZ-VAL, I. & HANSEN, C. (2017). Program evaluation and causal inference with high-dimensional data. *Econometrica* **85**, 233–98.

BICKEL, P. J., KLAASSEN, C. A., BICKEL, P. J., RITOV, Y., KLAASSEN, J., WELLNER, J. A. & RITOV, Y. (1993). *Efficient and Adaptive Estimation for Semiparametric Models*. Baltimore, Maryland: Johns Hopkins University Press.

BREIMAN, L. (1996). Stacked regressions. *Mach. Learn.* **24**, 49–64.

CAPONNETTO, A. & DE VITO, E. (2007). Optimal rates for the regularized least-squares algorithm. *Foundat. Comp. Math.* **7**, 331–68.

CHEN, T. & GUESTRIN, C. (2016). XGBoost: A scalable tree boosting system. In *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'16)*. New York: Association for Computing Machinery, pp. 785–94.

CHERNOZHUKOV, V., CHETVERIKOV, D., DEMIRER, M., DUFLO, E., HANSEN, C., NEWEY, W. & ROBINS, J. (2018). Double/debiased machine learning for treatment and structural parameters. *Economet. J.* **21**, C1–68.

CUCKER, F. & SMALE, S. (2002). On the mathematical foundations of learning. *Bull. Am. Math. Soc.* **39**, 1–49.

DING, P., FELLER, A. & MIRATRIX, L. (2019). Decomposing treatment effect variation. *J. Am. Statist. Assoc.* **114**, 304–17.

DORIE, V., HILL, J., SHALIT, U., SCOTT, M. & CERVONE, D. (2019). Automated versus do-it-yourself methods for causal inference: Lessons learned from a data analysis competition. *Statist. Sci.* **34**, 43–68.

DUDÍK, M., LANGFORD, J. & LI, L. (2011). Doubly robust policy evaluation and learning. In *Proc. 28th Int. Conf. Machine Learning (ICML'11)*. Madison, Wisconsin: Omnipress, pp. 1097–104.

FERWERDA, J., HAINMUELLER, J. & HAZLETT, C. J. (2017). Kernel-based regularized least squares in R (KRLS) and Stata (krls). *J. Statist. Software* **79**, 1–26.

FOSTER, D. J. & SYRGKANIS, V. (2019). Orthogonal statistical learning. *arXiv:* 1901.09036v2.

FRIEDMAN, J., HASTIE, T. & TIBSHIRANI, R. (2010). Regularization paths for generalized linear models via coordinate descent. *J. Statist. Software* **33**, 1–22.

FRIEDMAN, J. H. (1991). Multivariate adaptive regression splines. *Ann. Statist.* **19**, 1–67.

HAHN, P. R., MURRAY, J. S. & CARVALHO, C. M. (2020). Bayesian regression tree models for causal inference: Regularization, confounding, and heterogeneous effects. *Bayesian Anal.* **15**, 965–1056.

HILL, J. L. (2011). Bayesian nonparametric modeling for causal inference. *J. Comp. Graph. Statist.* **20**, 217–40.

HIRANO, K. & PORTER, J. R. (2009). Asymptotics for statistical treatment rules. *Econometrica* **77**, 1683–701.

IMAI, K. & RATKOVIC, M. (2013). Estimating treatment effect heterogeneity in randomized program evaluation. *Ann. Appl. Statist.* **7**, 443–70.

IMBENS, G. W. & RUBIN, D. B. (2015). *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. Cambridge: Cambridge University Press.

KENNEDY, E. H. (2020). Optimal doubly robust estimation of heterogeneous causal effects. *arXiv:* 2004.14497v2.

KITAGAWA, T. & TETENOV, A. (2018). Who should be treated? Empirical welfare maximization methods for treatment choice. *Econometrica* **86**, 591–616.

KÜNZEL, S. R., SEKHON, J. S., BICKEL, P. J. & YU, B. (2019). Metalearners for estimating heterogeneous treatment effects using machine learning. *Proc. Nat. Acad. Sci.* **116**, 4156–65.

LABER, E. & ZHAO, Y. (2015). Tree-based methods for individualized treatment regimes. *Biometrika* **102**, 501–14.

LUEDTKE, A. R. & VAN DER LAAN, M. J. (2016a). Optimal individualized treatments in resource-limited settings. *Int. J. Biostatist.* **12**, 283–303.

LUEDTKE, A. R. & VAN DER LAAN, M. J. (2016b). Statistical inference for the mean outcome under a possibly non-unique optimal treatment strategy. *Ann. Statist.* **44**, 713–42.

LUEDTKE, A. R. & VAN DER LAAN, M. J. (2016c). Super-learning of an optimal dynamic treatment rule. *Int. J. Biostatist.* **12**, 305–32.

MENDELSON, S. & NEEMAN, J. (2010). Regularization in kernel learning. *Ann. Statist.* **38**, 526–65.

NEWEY, W. K. (1994). The asymptotic variance of semiparametric estimators. *Econometrica* **62**, 1349–82.

Neyman, J. (1923). Sur les applications de la théorie des probabilités aux experiences agricoles: Essai des principes. *Roczniki Nauk Rolniczych* **10**, 1–51.
Obermeyer, Z. & Emanuel, E. J. (2016). Predicting the future—big data, machine learning, and clinical medicine. *New Engl. J. Med.* **375**, 1216–9.
Powers, S., Qian, J., Jung, K., Schuler, A., Shah, N. H., Hastie, T. & Tibshirani, R. (2018). Some methods for heterogeneous treatment effect estimation in high dimensions. *Statist. Med.* **37**, 1767–87.
R Development Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0. http://www.R-project.org.
Robins, J. M. (2004). Optimal structural nested models for optimal sequential decisions. In *Proc. 2nd Seattle Sympos. Biostatistics*. New York: Springer, pp. 189–326.
Robins, J. M., Li, L., Mukherjee, R., Tchetgen Tchetgen, E. & van der Vaart, A. (2017). Minimax estimation of a functional on a structured high-dimensional model. *Ann. Statist.* **45**, 1951–87.
Robins, J. M. & Rotnitzky, A. (1995). Semiparametric efficiency in multivariate regression models with missing data. *J. Am. Statist. Assoc.* **90**, 122–9.
Robinson, P. M. (1988). Root-N-consistent semiparametric regression. *Econometrica* **56**, 931–54.
Rosenbaum, P. R. & Rubin, D. B. (1983). The central role of the propensity score in observational studies for causal effects. *Biometrika* **70**, 41–55.
Rubin, D. B. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies. *J. Educ. Psychol.* **66**, 688–701.
Scharfstein, D. O., Rotnitzky, A. & Robins, J. M. (1999). Adjusting for nonignorable drop-out using semiparametric nonresponse models. *J. Am. Statist. Assoc.* **94**, 1096–120.
Schick, A. (1986). On asymptotically efficient estimation in semiparametric models. *Ann. Statist.* **14**, 1139–51.
Shalit, U., Johansson, F. D. & Sontag, D. (2017). Estimating individual treatment effect: Generalization bounds and algorithms. *Proc. Mach. Learn. Res.* **70**, 3076–85.
Smale, S. & Zhou, D.-X. (2003). Estimating the approximation error in learning theory. *Anal. Appl.* **1**, 17–41.
Steinwart, I. & Christmann, A. (2008). *Support Vector Machines*. New York: Springer.
Steinwart, I., Hush, D. R. & Scovel, C. (2009). Optimal rates for regularized least squares regression. In *22nd Conference on Learning Theory*. Montreal, Quebec, Canada, June 18–21, 2009.
Su, X., Tsai, C.-L., Wang, H., Nickerson, D. M. & Li, B. (2009). Subgroup analysis via recursive partitioning. *J. Mach. Learn. Res.* **10**, 141–58.
Talagrand, M. (2006). *The Generic Chaining: Upper and Lower Bounds of Stochastic Processes*. Berlin: Springer.
Tian, L., Alizadeh, A. A., Gentles, A. J. & Tibshirani, R. (2014). A simple method for estimating interactions between a treatment and a large number of covariates. *J. Am. Statist. Assoc.* **109**, 1517–32.
Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. R. Statist. Soc.* B **58**, 267–88.
Tsiatis, A. (2007). *Semiparametric Theory and Missing Data*. New York: Springer.
van der Laan, M. J. & Dudoit, S. (2003). Unified cross-validation methodology for selection among estimators and a general cross-validated adaptive epsilon-net estimator: Finite sample oracle inequalities and examples. Tech. rep. UC Berkeley Division of Biostatistics Working Paper Series No. 130.
van der Laan, M. J., Dudoit, S. & van der Vaart, A. W. (2006). The cross-validated adaptive epsilon-net estimator. *Statist. Decis.* **24**, 373–95.
van der Laan, M. J., Polley, E. C. & Hubbard, A. E. (2007). Super learner. *Statist. Applic. Genet. Molec. Biol.* **6**, article no. 25.
van der Laan, M. J. & Rose, S. (2011). *Targeted Learning: Causal Inference for Observational and Experimental Data*. New York: Springer.
van der Laan, M. J. & Rubin, D. (2006). Targeted maximum likelihood learning. *Int. J. Biostatist.* **2**, issue 1, article no. 11.
Wager, S. (2020). Cross-validation, risk estimation, and model selection: Comment on a paper by Rosset and Tibshirani. *J. Am. Statist. Assoc.* **115**, 157–60.
Wager, S. & Athey, S. (2018). Estimation and inference of heterogeneous treatment effects using random forests. *J. Am. Statist. Assoc.* **113**, 1228–42.
Wager, S., Du, W., Taylor, J. & Tibshirani, R. J. (2016). High-dimensional regression adjustments in randomized experiments. *Proc. Nat. Acad. Sci.* **113**, 12673–8.
Wolpert, D. H. (1992). Stacked generalization. *Neural Networks* **5**, 241–59.
Yang, Y. (2007). Consistency of cross validation for comparing regression procedures. *Ann. Statist.* **35**, 2450–73.
Zhang, B., Tsiatis, A. A., Davidian, M., Zhang, M. & Laber, E. (2012). Estimating optimal treatment regimes from a classification perspective. *Stat.* **1**, 103–14.
Zhao, Q., Small, D. S. & Ertefaie, A. (2018). Selective inference for effect modification via the lasso. *arXiv:* 1705.08020v3.

[*Received on* 4 *February* 2019. *Editorial decision on* 5 *August* 2020]