

University of California, Berkeley
U.C. Berkeley Division of Biostatistics Working Paper Series

Year 2011

Paper 286

Variable Importance Analysis with the
multiPIM R Package

Stephan J. Ritter*

Nicholas P. Jewell[†]

Alan E. Hubbard[‡]

*Division of Biostatistics, University of California, Berkeley, stephanritterRpacks@gmail.com

[†]Division of Biostatistics, University of California, Berkeley, jewell@berkeley.edu

[‡]Division of Biostatistics, University of California, Berkeley, hubbard@berkeley.edu

This working paper is hosted by The Berkeley Electronic Press (bepress) and may not be commercially reproduced without the permission of the copyright holder.

<http://biostats.bepress.com/ucbbiostat/paper286>

Copyright ©2011 by the authors.

Variable Importance Analysis with the multiPIM R Package

Stephan J. Ritter, Nicholas P. Jewell, and Alan E. Hubbard

Abstract

We describe the R package multiPIM, including statistical background, functionality and user options. The package is for variable importance analysis, and is meant primarily for analyzing data from exploratory epidemiological studies, though it could certainly be applied in other areas as well. The approach taken to variable importance comes from the causal inference field, and is different from approaches taken in other R packages. By default, multiPIM uses a double robust targeted maximum likelihood estimator (TMLE) of a parameter akin to the attributable risk. Several regression methods/machine learning algorithms are available for estimating the nuisance parameters of the models, including super learner, a meta-learner which combines several different algorithms into one. We describe a simulation in which the double robust TMLE is compared to the graphical computation estimator. We also provide example analyses using two data sets which are included with the package.

1. Introduction

1.1. Motivation

In most observational epidemiological studies, the investigators are interested in determining the independent association between one or more exposure variables and an outcome of interest, which requires adjustment for potentially confounding variables. Typically, and especially when the study is one of the first to be done in a certain field, there is little or no *a priori* knowledge of which variables are “important.” Thus, a wide net is cast, many variables are measured, and a multivariate regression model is built in an effort to adjust for confounding. The supposed effects of each exposure variable are read off from the estimated coefficients of this model, and statistical inference (confidence intervals or *p* values) is based on the standard error associated with each coefficient.

However, the interpretation of the model coefficients as the causal effects, and the inference obtained, are only valid if the model used in the analysis 1) closely approximates the true model and 2) has been specified *a priori*, without any feedback from the data. Unfortunately, these two conditions are somewhat mutually exclusive. Since there is rarely any real, *a priori*, knowledge of the true form of the relationships between the variables, the only way that one can hope to specify the model accurately is through a feedback cycle of multiple candidate models which are tested against the data and modified accordingly. Usually only a very small space of possible models can be searched by this *ad hoc* method. Thus, there is not much chance that the final model selected will closely approximate the true model. An incorrectly specified model will result in biased estimates of the adjusted associations. Failure to account for using feedback from the data in the analysis will typically result in artificially low standard errors. The likely result is a misleading, narrow confidence interval (or low *p*-value) around a biased estimate.

Even more problematic is the fact that the parameter used may depend on the model selected, and thus on the data. For example, for one realization of the data, the informal model selection may result in no interaction terms with the variable of interest, and thus only one measure of association would be reported. However, for another realization of the data, the procedure may choose a multiplicative interaction term in the final model, and in this case the researcher would report two measures of the adjusted association, for the two different levels of the presumed effect modifier. In this scenario, the concept of a sampling distribution of an estimator, which forms the basis for inference, breaks down.

[van der Laan \(2006\)](#) has proposed a solution to the problems outlined above. He takes a causal inference approach, and suggests that the first step is to decide on a real-valued parameter of interest (the variable importance measure). Then an efficient and robust estimator of this parameter can be derived using estimating equation methodology ([van der Laan and Robins 2003](#)) or targeted maximum likelihood ([van der Laan and Rubin 2006](#); [van der Laan and Rose 2011](#)). This estimator is then applied to each variable in turn, instead of estimating a single, global regression model.

The whole data analysis can be specified *a priori*, and one can still optimize the model selection by making aggressive use of modern machine learning algorithms to estimate the nuisance parameters of the model. These algorithms will data-adaptively search a large model space, and thereby, hopefully, come up with a reasonable approximation to the true data-generating distribution. By turning this entire multi-step procedure into an *a priori* specified black box,

one can harness the power of aggressive computing to obtain consistent estimates with honest inference.

Other approaches to variable importance analysis are provided by R packages such as **randomForest** (Liaw and Wiener 2002) and **caret** (Kuhn *et al.* 2011; Kuhn 2008). While these approaches also make use of modern machine learning methods, most of them are based on assessing changes in the risk function, while the approach of van der Laan (2006) is a type of adjusted mean inspired by causal inference (though the general targeted maximum likelihood approach can also be adapted to estimation of the risk function itself). This has several additional advantages that are not shared by these other approaches. For example, the choice of the parameter of interest can be tailored to the specific problem, and since the method is not tied to a specific learning algorithm, one can combine several arbitrary algorithms in a super learner in order to estimate nuisance parameters (see Section 2).

1.2. Overview

In this report, we introduce the package **multiPIM**, written for the R statistical computing environment (R Development Core Team 2011), and available for download from the Comprehensive R Archive Network (CRAN, <http://cran.r-project.org/package=multiPIM>)¹. **multiPIM** performs variable importance analysis by fitting multiple Population Intervention Models (PIMs, Hubbard and van der Laan 2008) to a data set with possibly many exposures (or treatments) of interest and possibly many outcomes of interest. In Section 2, we summarize the statistical properties of PIMs, describing in particular the procedure for their estimation using data-adaptive machine learning algorithms. In Section 3, we go into more detail about **multiPIM** and its functions. Section 4 describes a simulation which demonstrates the benefit of using a double robust estimator. In Section 5 and Section 6, we report on reanalyses of data from the Western Collaborative Group Study (WCGS, Rosenman *et al.* 1966, 1975) and from a study of water contact and schistosomiasis infection (Spear *et al.* 2004; Sudat *et al.* 2010). Section 6 (schistosomiasis example) includes the code for running the analysis. We close with a discussion in Section 7.

2. Statistical methodology

The approach for estimating PIMs was presented by Hubbard and van der Laan (2008), and is also described in Young (2007) and Young *et al.* (2009). We summarize the main points here.

2.1. Data structure and parameter of interest

Let us assume that we have an observed data structure

$$O = (Y, A, W) \sim P_0,$$

where Y is an outcome (which may be either binary or continuous), A is a binary exposure or treatment variable, W may be a vector of possible confounders of the effect of A on Y , and P_0 is the data-generating distribution. The parameter of interest is

¹As of this writing, the current version is 1.3-1

$$\begin{aligned}\psi(P_0) = \psi &= E_W[E(Y|A = 0, W) - E(Y|W)] \\ &= E_W[E(Y|A = 0, W)] - E(Y).\end{aligned}$$

This parameter is a type of attributable risk; it compares the overall mean of the outcome to the mean for a target group (defined by $A = 0$), averaged over strata of W . The hypothetical full data is given by

$$X = (Y_0, Y, W) \sim P_X,$$

where Y_0 is the counterfactual outcome for $A = 0$, i.e., the outcome as it would be under universal application of treatment $A = 0$. The causal analogue to $\psi(P_0)$ is

$$\begin{aligned}\psi(P_X) &= E_W[E(Y_0|W) - E(Y|W)] \\ &= E[Y_0 - Y].\end{aligned}$$

There are certain assumptions under which $\psi(P_0) = \psi(P_X)$. The assumptions are:

1. Time ordering assumption: W preceded A and A preceded Y in time. More generally, the data-generating distribution conforms to a specific nonparametric structural equation model (Pearl 2000).
2. Consistency assumption: The observed data structure, O , is a missing data structure on X (van der Laan and Robins 2003).
3. There is no unmeasured confounding, or, equivalently, A is conditionally independent of Y_0 , given W (van der Laan and Robins 2003).
4. The experimental treatment assignment (ETA) assumption or positivity assumption: the probability that A is zero, given W , is bounded away from zero, or $Pr(A = 0|W) > 0$. This is a relaxed version of the ETA assumption, which for certain other parameters requires a positive probability of having *each* of the treatment levels over the distribution of W in the target population (van der Laan and Robins 2003; Messer *et al.* 2010).

If these four assumptions hold, and the relevant models are estimated consistently, ψ can be thought of as an actual causal effect of A on Y . More specifically, it can be thought of as measuring the hypothetical effect of an intervention in which everyone in the population is made to be like the members of the target group. For example, if the target group corresponds to “unexposed”, then ψ would be the effect, on the overall mean of the outcome, of eliminating the exposure entirely. However, even if some of these assumptions do not hold, ψ may still be an interesting and worthwhile parameter to pursue. In this case, it can still be thought of as a type of variable importance (van der Laan 2006), and thus as a good way of quantifying the (W -adjusted) level of association between A and Y .

The term attributable risk has had several slightly different meanings in the epidemiological literature, and some authors prefer the term attributable fraction (e.g., Greenland and

Drescher 1993). For the case of a binary outcome, our parameter, ψ , is like a causal (W -adjusted) and sign-reversed version of what Gordis (2004) calls the “attributable risk in the total population”:

$$\left(\begin{array}{c} \text{Incidence in} \\ \text{total population} \end{array} \right) - \left(\begin{array}{c} \text{Incidence in} \\ \text{nonexposed group} \\ \text{(background risk)} \end{array} \right) \quad (1)$$

(Gordis 2004, Formula 12.3). In Section 5 below we will calculate a “naive attributable risk,” (naive since it is bivariate only and does not account for confounding) as the difference between the mean of the binary coronary heart disease outcome for an unexposed group and the overall mean of the outcome.

The goal of this work is to create an automated algorithm for estimating ψ for data sets with potentially many outcomes (Y 's) and many exposures (A 's) of interest, as well as potentially high dimensional W .

2.2. Estimators

Two general classes of estimators are available for estimating ψ : plug-in maximum likelihood-type estimators and estimating equation estimators. In the **multiPIM** package, we have implemented two estimators from each class. The estimator to be used is specified by supplying the `estimator` argument to the `multiPIM` or `multiPIMboot` functions.

Estimating equation approaches

The estimating equation estimators available in the **multiPIM** package are the inverse-probability-of-censoring-weighted (IPCW) estimator, and its doubly-robust extension (DR-IPCW). These estimators are derived in Hubbard and van der Laan (2008), and the derivations are based on the approach described in van der Laan and Robins (2003).

Let O_1, O_2, \dots, O_n be independent and identically distributed observations of O , with $O_i = (Y_i, A_i, W_i)$, $i \in \{1, 2, \dots, n\}$. An IPCW estimator is given by

$$\hat{\psi}_n^{IPCW} = \frac{1}{n} \sum_{i=1}^n \left[\left(\frac{I(A_i = 0)}{g_n(0|W_i)} - 1 \right) Y_i \right], \quad (2)$$

and the corresponding DR-IPCW estimator is given by

$$\hat{\psi}_n^{DR-IPCW} = \frac{1}{n} \sum_{i=1}^n \left[\left(\frac{I(A_i = 0)}{g_n(0|W_i)} - 1 \right) Y_i - \left(\frac{I(A_i = 0)}{g_n(0|W_i)} - 1 \right) Q_n(0, W_i) \right]. \quad (3)$$

$g_n(0|W)$ and $Q_n(0, W)$ are estimates of the nuisance parameters, $g(0|W)$ and $Q(0, W)$, respectively. $g(a|W)$ is the probability of having treatment level $A = a$ given covariates W (also known as the treatment mechanism or propensity score), and thus $g(0|W)$ is the probability of being in the target treatment group (the group defined by $A = 0$) given observed covariates W , or $P(A = 0|W)$. Similarly, $Q(a, W)$ is the mean value of the outcome, Y , given treatment level $A = a$ and covariates W , and thus $Q(0, W)$ is the mean value of Y , given treatment level $A = 0$, and given the observed level of covariates W , or $E[Y|A = 0, W]$.

These nuisance parameters usually need to be estimated from the data. Since A is binary, $g(0|W)$ can be estimated using some form of regression with which one can predict the class probabilities for a binary outcome. The estimate, $g_n(0|W)$, is taken as the predicted probability of being in the class given by $A = 0$, for a subject with covariates W . $Q(0, W)$ can be estimated by regressing Y on A and W . The estimate, $Q_n(0, W)$, can be found by using this regression model to predict on a new data set for which every element of A is set to zero, but W stays the same. The type of regression which should be used for building this model depends on whether Y is a binary or continuous variable.

Plug-in estimators

multiPIM also makes available two plug-in estimators: the graphical computation (G-computation) estimator (Robins 1986, 2000; van der Laan and Rubin 2006), and the targeted maximum likelihood estimator (TMLE, van der Laan and Rubin 2006; van der Laan and Rose 2011).

The G-computation estimator is given by

$$\begin{aligned}\hat{\psi}^{G-COMP} &= \hat{E}[Y_0] - \hat{E}[Y] \\ &= \frac{1}{n} \sum_{i=1}^n [Q_n^0(0, W_i)] - \bar{Y}.\end{aligned}$$

It is referred to as "G-COMP" in the package (e.g., `estimator = "G-COMP"`). Greenland and Drescher (1993) proposed an estimator that would encompass this parameter in the context of parametric logistic regression.

For the case of continuous Y , the TMLE is given by

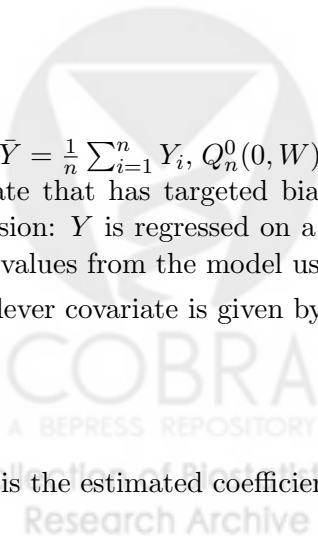
$$\begin{aligned}\hat{\psi}^{TMLE} &= \hat{E}[Y_0] - \hat{E}[Y] \\ &= \frac{1}{n} \sum_{i=1}^n [Q_n^1(0, W_i)] - \bar{Y} \\ &= \frac{1}{n} \sum_{i=1}^n [Q_n^0(0, W_i) + \hat{\epsilon}Z(0, W)] - \bar{Y}.\end{aligned}$$

Here, $\bar{Y} = \frac{1}{n} \sum_{i=1}^n Y_i$, $Q_n^0(0, W)$ is an initial estimate of $Q(0, W)$, and $Q_n^1(0, W)$ is an updated estimate that has targeted bias reduction for ψ . The updating is done by fitting a linear regression: Y is regressed on a "clever covariate" with no intercept and with $Q_n^0(A, W)$ (the fitted values from the model used for $Q(0, W)$) as offset (Rose and van der Laan 2011a).

The clever covariate is given by

$$Z(A, W) = \frac{I(A = 0)}{g_n(A|W)}$$

and $\hat{\epsilon}$ is the estimated coefficient for this covariate.



For the case of binary Y , the updating is done by logistic regression, the offset is $\text{logit}(Q_n^0(A, W))$, and

$$\hat{E}[Y_0] = \frac{1}{n} \sum_{i=1}^n \text{expit}(\text{logit}(Q_n^0(0, W_i)) + \hat{\epsilon}Z(0, W)).$$

where expit refers to the inverse logit function:

$$\text{expit}(\alpha) = \text{logit}^{-1}(\alpha) = \frac{1}{1 + e^{-\alpha}}.$$

An article about another R package for targeted maximum likelihood estimation, **tmle**, has recently appeared (Gruber and van der Laan 2012). This package is also available on CRAN, and implements a TMLE for the marginal additive effect of a binary point treatment. It also calculates estimates of the risk ratio and odds ratio for binary outcomes, and can be used to estimate controlled direct effects and the parameter of a marginal structural model.

2.3. Properties of the estimators

The IPCW estimator will be consistent if $g(0|W)$ is estimated consistently, and the G-Computation estimator will be consistent if $Q(0, W)$ is estimated consistently. The DR-IPCW estimator and the TMLE are both based on the efficient influence curve for the semiparametric model, and as a result, they have the double robustness property (meaning that they will be consistent if either $g(0|W)$ or $Q(0, W)$ is estimated consistently), and they are asymptotically locally efficient (Hubbard and van der Laan 2008; van der Laan and Robins 2003; van der Laan and Rose 2011). As a plug-in estimator, the TMLE has the advantage that the parameter estimate is guaranteed to fall in the natural range, assuming that an appropriate estimate of $Q(0, W)$ is used. For additional advantages of TMLE over other estimators, see Rose and van der Laan (2011b).

2.4. Super learner and recommendations for estimation of nuisance parameters

Since the consistency of the estimators is dependent upon consistent estimation of the nuisance parameters, it is worthwhile to devote some statistical and computational effort to this estimation. Thus, the **multiPIM** package makes use of the super learner (Sinisi *et al.* 2007; van der Laan *et al.* 2007; Polley *et al.* 2011; see also Breiman 1996). The super learner is a data-adaptive meta learner which can be used to do prediction of binary or continuous outcomes. The form of the super learner implemented in the **multiPIM** package uses v -fold cross-validation to select the best from among a set of “candidate learners” (Sinisi *et al.* 2007). This form is also known as the “discrete” super learner. In the more recent super learner algorithm, the final predictor is a weighted combination of the learners in the “library” (Polley *et al.* 2011). The candidate learners may normally be any arbitrary regression method or machine learning algorithm which maps the values of a set of predictors into an outcome value. In the **multiPIM** package, we have implemented a small set of candidates. Most of these rely on separate R packages, which are also available on CRAN. The user can select

from among these candidates in building a super learner to estimate $g(0|W)$ or $Q(0, W)$. This will be described in greater detail in Section 3.

Theoretical results have shown that, asymptotically, the super learner performs as well as the so-called oracle selector, which always chooses the best-performing candidate (the level of performance of a candidate is measured by a specific loss function; [Sinisi et al. 2007](#); [van der Laan et al. 2007](#)). Since it is usually the case that the data-generating distributions are unknown, combining many different candidates using a super learner (and thereby searching a very large model space) should reduce the bias of $g_n(0|W)$ and $Q_n(0, W)$. Thus we recommend using the super learner for estimation of nuisance parameters. However, there is an exception to this rule when using the TMLE. As a consequence of the bias reduction step (the updating of $Q_n(0, W)$ with the clever covariate), using a very aggressive algorithm to estimate $g(0|W)$ may cause empirical ETA violations, which could result in biased parameter estimates ([Petersen et al. 2011](#)). Thus super learning is not recommended for estimating $g(0|W)$ when using the TMLE. Since TMLE is the default estimator in the **multiPIM** package, we have set the default method for estimating $g(0|W)$ to be main terms logistic regression (see Section 3).

A more elegant solution to this problem is to use a sequence of increasingly non-parametric estimates of $g(0|W)$. This is the collaborative targeted maximum likelihood estimator ([van der Laan and Gruber 2010](#); [Gruber and van der Laan 2011](#)). However, this estimator has not yet been implemented for the population intervention parameter we are using here.

2.5. Inference

One can show that the four estimators described above are asymptotically linear, with asymptotically normal sampling distributions under very general conditions ([van der Laan and Robins 2003](#); [Zheng and van der Laan 2011](#)). Two methods of estimating the variance are available in the **multiPIM** package. The “plug-in” estimates are based on the influence curve ([van der Laan and Robins 2003](#); this method is not available for the G-Computation estimator). Specifically, if $IC(O; \hat{P})$ is the plug-in estimate of the influence curve (where estimates of the relevant parts of P_0 are represented by \hat{P}), then the plug-in standard error is

$$\hat{\sigma}^{plug-in} = \sqrt{\frac{\widehat{Var}(IC(O; \hat{P}))}{n}}.$$

For example, note from Equation 2 and Equation 3 in Section 2.2, that the IPCW and DR-IPCW estimators are both expressed as means over a certain vector. To get the plug-in standard error for each of these estimators, take the standard deviation over this vector instead of the mean, and divide by \sqrt{n} .

The preferred method for estimating the variance is to use the bootstrap ([Efron 1979](#)). The bootstrap method is more robust, however it of course requires much more computation time. We note that the plug-in (influence curve-based) estimates of the variance for the IPCW estimator in particular tend to be overly conservative ([van der Laan et al. 2003](#)).

3. The multiPIM R Package

The **multiPIM** package consists of two main functions, two methods, and four character vectors. The **multiPIM** function provides the main variable importance analysis functionality.

The `multiPIMboot` function can be used to bootstrap the `multiPIM` function and get bootstrap standard errors for the parameter estimates. There is a summary method for the class "multiPIM" objects which are returned by the functions `multiPIM` and `multiPIMboot`, and a print method for the summary objects. Finally, the elements of the four character vectors (`all.bin.cands`, `default.bin.cands`, `all.cont.cands` and `default.cont.cands`) represent the regression methods/machine learning algorithms which are available for estimating the nuisance parameters $g(0|W)$ and $Q(0, W)$ (see Section 3.5 and Section 3.6), and are meant to be passed in as arguments to `multiPIM` and `multiPIMboot`.

3.1. Input and output

The arguments to the `multiPIM` function are as follows:

```
multiPIM(Y, A, W = NULL,
         estimator = c("TMLE", "DR-IPCW", "IPCW", "G-COMP"),
         g.method = "main.terms.logistic", g.sl.cands = NULL,
         g.num.folds = NULL, g.num.splits = NULL,
         Q.method = "sl", Q.sl.cands = "default",
         Q.num.folds = 5, Q.num.splits = 1,
         Q.type = NULL,
         adjust.for.other.As = TRUE,
         truncate = 0.05,
         return.final.models = TRUE,
         na.action,
         check.input = TRUE,
         verbose = FALSE,
         extra.cands = NULL,
         standardize = TRUE,
         ...)
```

The main input to the `multiPIM` function is in the form of three data frames: W , A and Y . Each of these data frames may contain multiple columns. The data frame A should contain binary (0/1) exposure variables, and Y should contain outcome variables. W is optional. If supplied, it should contain covariate variables which the user wishes to include in the adjustment set, but for which he/she is not interested in estimating a variable importance. The function calculates one estimate of ψ for each exposure-outcome pair. That is, if $A^{(j)}$ is the j th of J exposure variables (i.e., the j th of J columns of A) and if $Y^{(k)}$ is the k th of K outcome variables (i.e., the k th of K columns of Y), then JK estimates, $\hat{\psi}_{j,k}$, of ψ will be calculated, one for each pair $(A^{(j)}, Y^{(k)})$ such that $j \in \{1, 2, \dots, J\}$ and $k \in \{1, 2, \dots, K\}$.

3.2. Adjustment Set and the `adjust.for.other.As` argument

With the `adjust.for.other.As` argument, the user can control which variables are kept in the adjustment set in calculating the estimate, $\hat{\psi}_{j,k}$, for each pair, $(A^{(j)}, Y^{(k)})$. If `adjust.for.other.As` is `TRUE`, the other columns of the data frame A , i.e., all $A^{(j^*)}$ such that $j^* \neq j$, will be included in the adjustment set. That is, in the notation of Section 2, they will be included as members of W , and thus will be included, along with the columns of the

data frame W , as possible covariates to select from in building the models from which $g(0|W)$ and $Q(0, W)$ are estimated. If `adjust.for.other.As` is set to `FALSE`, the other columns of A will not be included in the adjustment set. In this case, the data frame W must be supplied. If W is supplied, the variables it contains will be included in the adjustment set for all models, no matter the value of `adjust.for.other.As`.

3.3. Rebuilding of models used to estimate $Q(0, W)$

When **multiPIM** is run, it builds only one model per exposure variable (column of A), from which $g(0|W)$ is estimated. The estimate of $g(0|W)$ based on each model is then used in the calculation of each of the parameter estimates which involves the corresponding exposure variable. However, this is not the case for the models from which $Q(0, W)$ is estimated. The model for a specific outcome variable is rebuilt each time the effect of a new exposure variable is being calculated. One property of some of the machine learning algorithms used as candidates for the super learner is that they may drop certain covariate variables from the model. In order to ensure the smoothness of the sampling distribution of the estimator, the relevant exposure variable is forced back in to any model for $Q(0, W)$ from which it has been dropped (see Section 3.5 and Section 3.6 and see also [Zheng and van der Laan 2011](#)). Thus, one such model must be built per exposure-outcome pair.

3.4. Implementation of super learner

As mentioned in Section 2.4, the preferred method for estimating the nuisance parameters is to use a super learner with many candidates (with the exception of estimating $g(0|W)$ when using the TMLE). The implementation of the super learner in the **multiPIM** package uses v -fold cross-validation to select the best from among a set of candidate learners. All exposure variables in A must be binary, and thus only binary outcome regression methods are implemented for use in building a super learner to estimate $g(0|W)$. However, the outcome variables in Y may be binary or continuous, and thus some continuous outcome regression methods/machine learning algorithms are implemented as well, in order to be used in a super learner for estimating $Q(0, W)$. The performance of candidates in a binary outcome super learner is evaluated using the negative log likelihood loss function. The performance of candidates in a continuous outcome super learner is evaluated using the mean squared error loss function. The following two sections describe the default candidate algorithms which have been implemented.

3.5. Default binary outcome candidates

The default binary outcome super learner candidates are given by the vector `default.bin.cands`:

```
default.bin.cands <- c("polyclass", "penalized.bin", "main.terms.logistic")
```

The point of making this vector, and the vectors `all.bin.cands`, `default.cont.cands`, and `all.cont.cands` available to the user is so that they (or subsets of their elements) may be passed to the `multiPIM` and `multiPIMboot` functions as the arguments `g.method`, `Q.method`, `g.sl.cands` or `Q.sl.cands`. This is the mechanism whereby the user may specify which regression methods should be used in building models to estimate $g(0|W)$ and $Q(0, W)$.

Polyclass candidate

This super learner candidate uses the function `polyclass` from the R package **polyspline** (Kooperberg *et al.* 1997; Stone *et al.* 1997; Kooperberg 2010). `polyclass` fits linear splines and their tensor products using a model selection process guided by the Akaike information criterion (Kooperberg *et al.* 1997).

Since there is a possibility with this algorithm that certain variables may be dropped from the model, the implementation is slightly different for the case when this candidate is being used to estimate $Q(0, W)$ vs. when it is being used to estimate $g(0|W)$. In order to make sure that the relevant exposure variable stays in the model when estimating $Q(0, W)$ (see Section 3.3), the object returned by `polyclass` is inspected to see if the relevant variable is a member of the basis functions selected. If it is not, a second, logistic regression model is fit using the predictions from the `polyclass` model as a covariate, along with the relevant exposure variable which was dropped from the `polyclass` model. In order to stay somewhat flexible, an interaction term between the exposure variable and the predictions from `polyclass` is also included in this logistic model. Thus, this secondary logistic model has the form

$$\text{logit} \left(Q_k^*(A^{(j)}, W^*) \right) = \beta_0 + \beta_1 A^{(j)} + \beta_2 \text{logit} \left(\hat{Q}_k^0(A^{(j)}, W^*) \right) + \beta_3 A^{(j)} \text{logit} \left(\hat{Q}_k^0(A^{(j)}, W^*) \right),$$

where W^* depends on the value of the `adjust.for.other.As` argument, $\hat{Q}_k^0(A^{(j)}, W^*)$ are the predictions from a `polyclass` model for which the outcome was $Y^{(k)}$, and the β 's are regression coefficients. Note that this model contains the original `polyclass` model as a submodel (just set $\beta_2 = 1$ and $\beta_0, \beta_1, \beta_3 = 0$).

After both the `polyclass` and the logistic model have been fit, prediction on a new data set is done by first getting the `polyclass` model's predictions on this new data, and then predicting on these new predictions using the logistic model.

Penalized candidate

The second binary outcome candidate, named "penalized.bin" to distinguish it from the continuous outcome version, is based on the `penalized` function from the R package **penalized** (Goeman 2010, 2011). This function performs regressions with either L1 (lasso) or L2 (ridge) penalties, or with a combination of the two. The implementation of this candidate in the `multiPIM` function uses only an L1 penalty. The value of the penalty is chosen using the `profL1` function, also from package **penalized**. With `profL1`, cross-validation is carried out on ten possible values of the L1 penalty. The values range from zero to the minimum value which would cause all coefficients to shrink to zero.

When estimating $g(0|W)$, all columns of W (and, depending on `adjust.for.other.As`, possibly all other columns of A besides the one actually being modeled) are put into the `penalized` model as penalized main terms. However, when estimating $Q(0, W)$, in order to prevent shrinking of its coefficient to zero, the relevant exposure variable is added to the model as an unpenalized covariate.

Main terms logistic candidate

The final binary outcome candidate uses the function `glm` from the package **stats** (R Development Core Team 2011) to build a main terms logistic regression model. Since `glm` does not

drop any covariates, it is not necessary to change the implementation for when $Q(0, W)$ is being estimated vs. when $g(0|W)$ is being estimated.

3.6. Default continuous outcome candidates

The default continuous outcome super learner candidates are given by the vector `default.cont.cands`:

```
default.cont.cands <- c("polymars", "lars", "main.terms.linear")
```

Since the exposure variables are always binary, continuous outcome candidates are always used for estimating $Q(0, W)$, and never for $g(0|W)$. Thus, they must always allow for the forcing of variables into the model.

Polymars candidate

The `polymars` candidate uses the `polymars` function, from package **polspline**. This function is similar to the `polyclass` function, but it can be used for modeling a continuous outcome instead of a categorical outcome. Another difference between the two functions is that the `polymars` function has a mechanism for forcing specific variables into the model. This mechanism is used in the `polymars` candidate, and thus no extra regression model needs to be built in order to force the exposure variable back into the model in case it is dropped.

Lars candidate

This candidate is based on the function `lars` from the package **lars** (Efron *et al.* 2004; Hastie and Efron 2011). `lars` performs least angle regression, a variant of the lasso. For the `lars` candidate, the function `cv.lars` is used to cross-validate on a grid of possible points on the solution path. If the final `lars` model has a coefficient of 0 for the relevant exposure variable, a secondary linear regression model is fit using the exposure variable and the predictions from the `lars` model as covariates, similarly to the logistic regression model described above for the `polyclass` candidate (see Section 3.5.1).

Main terms linear candidate

The `"main.terms.linear"` candidate uses the function `lm` from the package **stats** to fit a main-terms-only linear regression model. Again, since no covariates are dropped by this method, no secondary forcing model is necessary.

3.7. Alternative regression methods and other user options

In addition to the default candidates mentioned above, there are several optional candidates which can be added to super learners. For both binary and continuous outcomes, there is a candidate based on the `rpart` function from package **rpart** (Therneau *et al.* 2010; Breiman 1984). There is also a continuous outcome version of the penalized candidate. The user of the `multiPIM` function may also use any of the super learner candidates mentioned as a stand-alone regression method for estimating $g(0|W)$ or $Q(0, W)$. This can be done by specifying the `g.method` or `Q.method` arguments as the the name of the desired candidate. The user may

also supply one or more arbitrary, self-implemented, regression methods, either to add to a super learner as candidates, or to use as stand-alone regression methods. This makes it possible to use the more recent version of the super learner, which has been implemented in the CRAN package **SuperLearner** (Polley and van der Laan 2011). Another recommended learner which can be added as a user-supplied candidate/method is the Deletion/Substitution/Addition algorithm (the **DSA** package, Neugebauer and Bullard 2010; Sinisi and van der Laan 2004). The mechanism for specifying user-supplied candidates/methods is via the `extra.cands` argument and is fully documented in the Candidates help file:

```
R> ?Candidates
```

The user may also choose:

- How many “folds” and splits to use for the v -fold cross-validation in the super learner (the defaults are 5 and 1, respectively)
- Whether and at which value to truncate (from below) $g_n(0|W)$ in order to avoid instability of the estimator (the default is to truncate at 0.05)

3.8. multiPIMboot function

The `multiPIMboot` function can be used instead of the `multiPIM` function when the user wishes to use bootstrapping to calculate standard errors. `multiPIMboot` will run `multiPIM` once on the original data set, then sample with replacement from the rows of the data and re-run `multiPIM` the desired number of times on resampled data sets. All arguments to `multiPIM` are also available for `multiPIMboot`, and these have the same defaults. Additional arguments are:

- `times`, for specifying the number of bootstrap replicates
- `id`, to identify clusters and perform a clustered bootstrap
- `multicore`, `mc.num.jobs`, and `mc.seed` for running the bootstrap on multiple processor cores from a single R session. This requires the **parallel** package, which is distributed with R as of version 2.14.0. Previous versions of **multiPIM** relied on the **multicore** and **rlecuyer** packages for this functionality (Urbanek 2011; Sevcikova and Rossini 2009).

Based on our experience with two different quad-core systems, the factor of speedup when `multicore = TRUE` is close to the number of (physical) cores used, which is not surprising since bootstrapping is embarrassingly parallel. Note that the single run of `multiPIM` on the original data is done first in serial, before **multicore**'s `parallel` function sends multiple bootstrap jobs to the cores. Thus the CPU usage will not hit 100% of all cores until the first run is complete.

In order to improve reproducibility of parallel runs when `multicore = TRUE`, the random number generator type is automatically set using

```
R> RNGkind("L'Ecuyer-CMRG")
```

This causes R to use an implementation of the generator described by L'Ecuyer *et al.* (2002), which allows using different and reproducible streams within each parallel thread of execution.

3.9. Statistical recommendations and effects on computation time

Most of the computation time spent when running the `multiPIM` function goes into fitting the models from which $g(0|W)$ and $Q(0, W)$ are estimated. Also, assuming the same options are used, the `multiPIMboot` function will take much longer to run than the `multiPIM` function, since it just repeatedly calls the `multiPIM` function. Thus, three very effective ways of reducing the computation time are to:

1. Use plug-in standard errors by running `multiPIM` instead of `multiPIMboot`
2. Use the IPCW estimator, which requires estimating only $g(0|W)$, and not $Q(0, W)$ (unlike the two double robust estimators, TMLE and DR-IPCW)
3. Do not use super learning, but instead use methods which require very little computation time, such as main terms linear and main terms logistic regression

However, for maximal robustness and accuracy of inference, it is recommended to use `multiPIMboot` with the default arguments (bootstrap standard errors, TMLE estimator, super learning to estimate $Q(0, W)$). The multicore functionality has been added as a way to reduce the time required to run this full bootstrap analysis.

Using the bootstrap instead of plug-in standard errors will have the greatest effect on running time. If bootstrap is just not feasible, it is recommended to run `multiPIM` with the TMLE estimator. Additional fine tuning of the running time can be done by using the `summary` method (see next section) to see which super learner candidates are using the most computation time. The computation time of the super learner depends on which candidates are included and on the number of splits and “folds” used for cross-validation. Using a single split and 5 folds should be adequate in most cases, but it may be safer to use a higher number of folds, such as 10, especially if the data set has only a few hundred observations. Increasing the number of splits beyond one may also improve the accuracy of the cross-validation candidate selection mechanism (Molinaro *et al.* 2005).

3.10. Summary method

Both the `multiPIM` and the `multiPIMboot` functions return objects of class “`multiPIM`”. We have written a summary method which can be called on these objects in order to generate numerical summaries of the statistical results (as well as a breakdown of where the computation time was spent). The method uses the parameter estimates and standard errors to calculate test statistics and unadjusted as well as Bonferroni-adjusted p values, and allows for easy and customizable printing of tables showing these results. We demonstrate this by running an example which has been adapted from the help file for the `multiPIM` function:

```
R> library("multiPIM")
R> num.columns <- 3
R> num.obs <- 250
R> set.seed(23)
```

We generate a data frame containing random binary data to use as exposure variables:

```
R> A <- as.data.frame(matrix(rbinom(num.columns*num.obs, 1, .5),
+                             nrow = num.obs, ncol = num.columns))
```

Next we generate outcomes based on the exposures, by starting with random noise and adding multiples of the exposure variables:

```
R> Y <- as.data.frame(matrix(rnorm(num.columns*num.obs),
+                             nrow = num.obs, ncol = num.columns))
R> for(i in 1:num.columns)
+   Y[, i] <- Y[, i] + i * A[, i]
```

Next we make sure that the two data frames have unique names, then run `multiPIM` on them and run `summary` on the resulting object:

```
R> names(A) <- paste("A", 1:num.columns, sep = "")
R> names(Y) <- paste("Y", 1:num.columns, sep = "")
R> result <- multiPIM(Y, A)
R> summary(result)
```

The call was:

```
multiPIM(Y = Y, A = A)
```

Results for the exposure "A1" vs the outcomes listed on the left:

| outcome | param.estimate | stand.error | test.stat | p.val | p.val.bon.adj |
|---------|----------------|-------------|-----------|-----------|---------------|
| Y1 | -0.57198 | 0.08182 | 6.9911 | 2.727e-12 | 2.455e-11 |
| Y2 | 0.02413 | 0.06810 | 0.3544 | 7.231e-01 | 1.000e+00 |
| Y3 | -0.03655 | 0.06482 | 0.5639 | 5.728e-01 | 1.000e+00 |

Results for the exposure "A2" vs the outcomes listed on the left:

| outcome | param.estimate | stand.error | test.stat | p.val | p.val.bon.adj |
|---------|----------------|-------------|-----------|-----------|---------------|
| Y1 | -0.008016 | 0.06724 | 0.1192 | 9.051e-01 | 1.000e+00 |
| Y2 | -0.935624 | 0.08708 | 10.7444 | 6.298e-27 | 5.668e-26 |
| Y3 | -0.029830 | 0.05764 | 0.5176 | 6.048e-01 | 1.000e+00 |

Results for the exposure "A3" vs the outcomes listed on the left:

| outcome | param.estimate | stand.error | test.stat | p.val | p.val.bon.adj |
|---------|----------------|-------------|-----------|-----------|---------------|
| Y1 | 0.04203 | 0.07485 | 0.5615 | 5.744e-01 | 1.000e+00 |
| Y2 | 0.11055 | 0.06871 | 1.6090 | 1.076e-01 | 9.685e-01 |
| Y3 | -1.57871 | 0.11371 | 13.8832 | 8.012e-44 | 7.211e-43 |

Total time for main multiPIM run:

4.035824 seconds

Breakdown by g vs. Q modeling:

| | method | seconds | %.of.total |
|------------|---------------------|---------|------------|
| g modeling | main.terms.logistic | 0.02838 | 0.7032 |
| Q modeling | sl | 3.97188 | 98.4156 |

Total time for Q model super learner cross validation (x-val):

3.744127 seconds

Breakdown by candidate:

| | seconds | %.of.Q.x-val.time |
|-------------------|---------|-------------------|
| polymars | 0.7125 | 19.03 |
| lars | 2.7289 | 72.89 |
| main.terms.linear | 0.2744 | 7.33 |

Notice that for the pairs $A[, i]$ vs. $Y[, i]$, $i = 1$ to 3, the adjusted p values get progressively lower, since $Y[, i]$ is $i * A[, i]$ plus noise. However, off-diagonal p values are higher since there is no dependence of $Y[, i]$ on $A[, j]$ when $i \neq j$. There is a corresponding trend in the actual parameter estimates, which get progressively more negative for the diagonal ($A[, i]$ vs. $Y[, i]$, $i = 1$ to 3) exposure-outcome pairs.


The breakdown of the computation time shows that most of the time goes into the super learning for the Q model, most of this Q modeling time is being spent on the lars candidate.

4. Simulation

In order to demonstrate the benefit of using a double robust estimator, we compared the TMLE to the G-Computation estimator in a simulation. The complete R script which runs the simulation can be found in the supplements to this paper.

4.1. Simulated Data

The data consisted of four covariate variables $W = (W^{(1)}, W^{(2)}, W^{(3)}, W^{(4)})$, a single exposure variable A and a single outcome variable Y . W was generated as multivariate normal with mean vector $\mu = (0, 0, 0, 0)$ and covariance matrix



$$\Sigma = \begin{pmatrix} 1 & 0.2 & 0.2 & 0.2 \\ 0.2 & 1 & 0.2 & 0.2 \\ 0.2 & 0.2 & 1 & 0.2 \\ 0.2 & 0.2 & 0.2 & 1 \end{pmatrix}$$

A was generated from a logistic model based on W , with

$$A_i \sim \text{Bernoulli}(\text{expit}(\beta W_i^T)),$$

where $\beta = (0.2, 0.2, 0.2, 0.2)$ and $W_i = (W_i^{(1)}, W_i^{(2)}, W_i^{(3)}, W_i^{(4)})$. We made the data generating model for Y more complex:

$$Y_i = W_i^{(1)}W_i^{(2)} + W_i^{(3)}W_i^{(4)} + A_i \left[(W_i^{(1)})^2 + (W_i^{(2)})^2 + (W_i^{(3)})^2 + (W_i^{(4)})^2 \right] + \text{error},$$

with the errors i.i.d. Normal($\mu = 0, \sigma^2 = 4$), and with $(W_i^{(j)})^2$ equal to the square of $W_i^{(j)}$, for $j = 1, 2, 3, 4$.

Data sets were generated with sample sizes, n , evenly spaced on the log scale between 100 and 250,000, with one data set per value of n .

4.2. Estimators

To get estimates of ψ , we ran the `multiPIM` function twice on each data set, once with `estimator = "TMLE"` and once with `estimator = "G-COMP"`. Nuisance parameters were estimated using main terms logistic regression for $g(0|W)$ (TMLE only) and main terms linear regression for $Q(0, W)$. Thus the model for $g(0|W)$ was correctly specified, while the model for $Q(0, W)$ was misspecified.

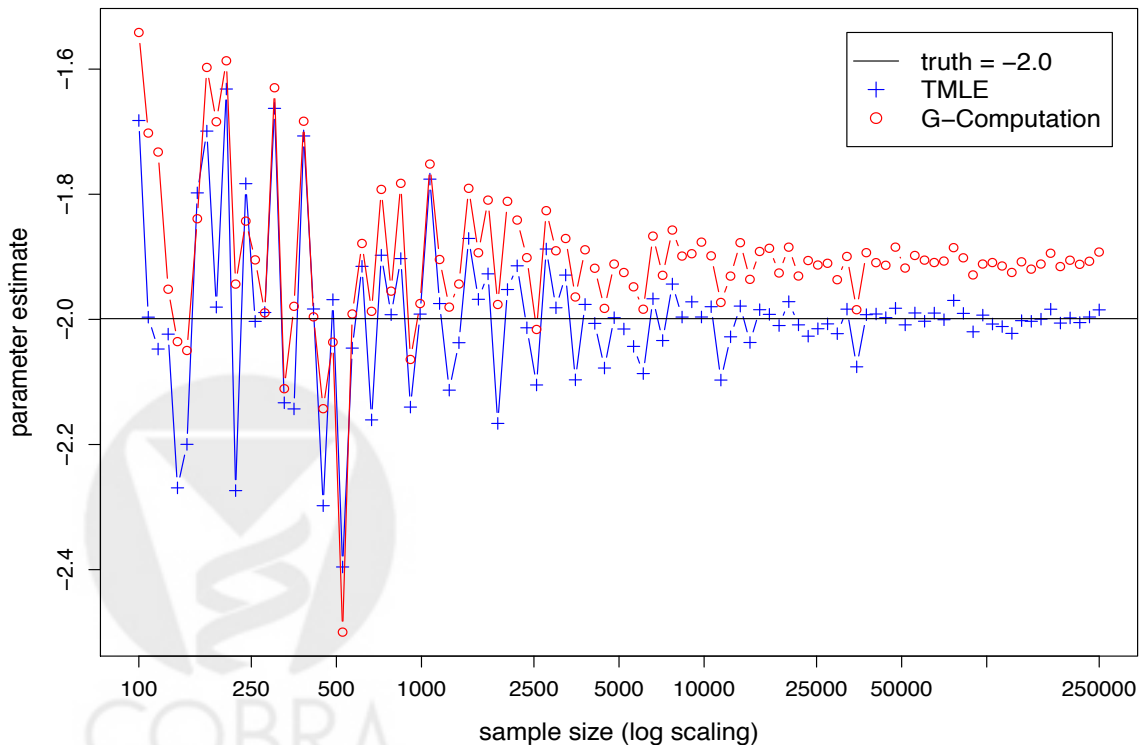


Figure 1: Simulation results. The TMLE estimates are centered around the true parameter value, while the G-Computation estimates are slightly biased.

4.3. Results

The results are shown in Figure 1. It is clear that there is some bias towards zero in the G-Computation estimates. This is due to the misspecified model for $Q(0, W)$. The TMLE estimator uses the same biased estimates of $Q(0, W)$ as the G-Computation estimator, but thanks to the double robustness and the fact that $g(0|W)$ is being estimated using the correct model, the TMLE estimates stay centered around the true parameter value. The variability of the two estimators appears to be similar throughout the range of sample sizes.

5. Analysis of data from the western collaborative group study

5.1. Study background

The Western Collaborative Group Study (WCGS) is a prospective study on coronary heart disease (CHD) which began in 1960 (Rosenman *et al.* 1966). The subjects were males, aged 39-59 at the start of follow-up, who were employed in several different corporations in California. The main goal of the study was to assess a possible effect of behavior type on incidence of CHD. After psychological testing, subjects were classified as having either of two behavior types, A or B. Type A is characterized by, among other qualities, “excessive drive, aggressiveness, and ambition”, while type B is characterized by having fewer of these qualities or having them in lesser extents (Rosenman *et al.* 1966).

5.2. Description of data

The data set has been included as part of the **multiPIM** package and can be loaded with

```
R> data("wcfgs")
```

The data analyzed represent a follow-up experience of 8.5 years (through 1969, Rosenman *et al.* 1975). The original subject enrollment count at the start of the study was 3524. However, subjects were later excluded for the following reasons: not being in the correct age range at intake, having CHD already manifest at intake, working for one specific corporation which pulled itself out of the study, or being lost to follow up for various reasons (Rosenman *et al.* 1975). This left 3154 subjects whose data was available for analysis. Since 12 of these subjects had missing values for one of the variables to be used in the analysis, these 12 subjects were also removed from the analysis for a final $n = 3142$. Of these final 3142 subjects, 257 (8.2%) had a CHD event.

Variables present in this data set include baseline covariates such as height and weight, the type A/B behavior pattern, total cholesterol levels, systolic and diastolic blood pressure, and smoking history (number of cigarettes smoked per day). Also present is the binary outcome variable indicating whether a CHD event occurred within the follow-up period.

The **multiPIM** function requires the exposure variables to be binary. Thus, they were dichotomized as follows:

- **typeAB**: This is just the behavior pattern variable with type A coded as 1 (exposed), and type B coded as 0 (unexposed). Thus the target group is type B behavior, in the

sense that the resulting parameter estimate, $\hat{\psi}$, can be thought of as an attributable risk which compares the level of the outcome in the entire population to the level for those with type B behavior.

- **chol**: The total cholesterol, dichotomized with a cutoff point of 240. The target group is therefore those with total cholesterol less than 240.
- **cigs**: This was coded as 1 for smokers (anyone who smokes 1 or more cigarettes per day) and 0 for non-smokers. Thus, the target group is non-smokers.
- **highBP**: Instead of having two highly correlated cholesterol variables, this single variable was coded as 1 (or exposed) if *either* diastolic blood pressure was greater than 90, or systolic blood pressure was greater than 140, and coded as 0 otherwise. Thus, the target group consists of individuals for whom neither measure is elevated.
- **bmi**: First the body mass index (BMI) was calculated by dividing weight in kg by the square of height in m². Then this variable was dichotomized using a cutoff point of BMI = 25. Anything greater than 25 was coded as 1 and anything less than 25 as 0. Thus, the target group is those with BMI less than 25.

These five variables were passed to the `multiPIMboot` function as the data frame **A**. Summary statistics for these exposure variables are given in Table 1.

| | unexposed group | proportion exposed | prop. unexp. with CHD | prop. exp. with CHD | naive attr. risk |
|--------|--------------------|-----------------------|--------------------------|------------------------|---------------------|
| typeAB | type B | 0.504 | 0.051 | 0.112 | -0.0311 |
| chol | < 240 | 0.343 | 0.057 | 0.129 | -0.0247 |
| cigs | < 1/day | 0.476 | 0.060 | 0.106 | -0.0223 |
| highBP | D<90 & S<140 | 0.212 | 0.070 | 0.126 | -0.0119 |
| bmi | < 25 | 0.411 | 0.073 | 0.095 | -0.0089 |

Table 1: Summary information for the five binary exposure variables used in the analysis. The first column states the characteristic that defines the unexposed group; D: diastolic blood pressure; S: systolic blood pressure; prop. unexp./exp. with CHD: proportion of unexposed/exposed with CHD; naive attr. risk: naive attributable risk – this is just the value in the 3rd column minus the overall disease rate of 0.082 (8.2%)

| | age | typeAB | chol | cigs | highBP | bmi | chd |
|--------|------|--------|------|------|--------|-------|------|
| age | 1.00 | 0.09 | 0.10 | 0.00 | 0.12 | 0.02 | 0.12 |
| typeAB | | 1.00 | 0.02 | 0.06 | 0.07 | 0.03 | 0.11 |
| chol | | | 1.00 | 0.08 | 0.10 | 0.04 | 0.12 |
| cigs | | | | 1.00 | -0.02 | -0.10 | 0.09 |
| highBP | | | | | 1.00 | 0.17 | 0.08 |
| bmi | | | | | | 1.00 | 0.04 |
| chd | | | | | | | 1.00 |

Table 2: Correlation matrix for all variables used in the analysis.

Also present in the data set was a variable giving the subjects' age in years. This variable was included in the analysis by being passed to `multiPIMboot` as the single-column data frame W . As stated above, ages ranged from 39-59 at the start of follow-up. Table 2 shows the correlation matrix for all variables used in the analysis.

5.3. Estimator used

The `multiPIMboot` function was run using the default estimator (TMLE) and the default methods for estimating nuisance parameters: main terms logistic regression for $g(0|W)$, super learning using the default binary outcome candidates (see Section 3.5) for the initial estimate of $Q(0, W)$. The `adjust.for.other.As` argument (see Section 3.2) was also kept at its default value of `TRUE`. The call to `multiPIMboot` was made as follows (for complete script see the supplements):

```
R> boot.result <- multiPIMboot(Y, A, W, times = 2000, multicore = TRUE,
+                               mc.num.jobs = 8, verbose = TRUE)
```

The elapsed time for running this job on a quad core iMac was about 4.6 hours.

5.4. Results

Table 3 shows the results of running the `multiPIMboot` function on the WCGS data. Three of the five exposure variables were found to have a significant effect on the CHD outcome after Bonferroni adjustment of p values. These were: A or B behavior type, cholesterol level, and cigarette smoking status. Since the outcome is binary, the parameter estimates (first column of Table 3), are on the scale of a proportion. For example, if one accepts the validity of the causal assumptions enumerated in Section 2.1, then the parameter estimate for the variable `typeAB` implies that if type A behavior were eliminated from the population and everyone was made to have behavior type B, the incidence of CHD would be reduced by about 2.7 percentage points (recall that 8.2% of the sample had an incident CHD event).

| | $\hat{\psi}^{TMLE}$ | $\hat{\sigma}^{plug-in}$ | $\hat{\sigma}^{boot}$ | T | p | $p^{Bonferroni}$ |
|--------|---------------------|--------------------------|-----------------------|------|----------|------------------|
| typeAB | -0.0271 | 0.0051 | 0.0051 | 5.33 | 9.66E-08 | 4.83E-07 |
| chol | -0.0201 | 0.0042 | 0.0042 | 4.78 | 1.76E-06 | 8.78E-06 |
| cigs | -0.0210 | 0.0048 | 0.0049 | 4.28 | 1.90E-05 | 9.50E-05 |
| highBP | -0.0074 | 0.0031 | 0.0030 | 2.43 | 0.015 | 0.076 |
| bmi | -0.0055 | 0.0044 | 0.0044 | 1.25 | 0.210 | 1.0 |

Table 3: Results of running the `multiPIMboot` function on the WCGS data. $\hat{\psi}^{TMLE}$: parameter estimates; $\hat{\sigma}^{plug-in}$: plug-in standard errors (from the influence curve); $\hat{\sigma}^{boot}$: bootstrap standard errors; T : test statistic (calculated using $\hat{\sigma}^{boot}$); p : unadjusted, two-sided p value from comparison of T with the standard normal distribution function; $p^{Bonferroni}$: Bonferroni-adjusted p value.

Comparison of Table 3 with Table 1 shows that the adjusted parameter estimates (first column of Table 3) are all less than the naive estimates (final column of Table 1). Some of the apparent

effect is being “adjusted out.” Also, the standard errors decrease with decreasing proportion exposed. For example, the lowest bootstrap standard error, 0.0030, is for the variable highBP, for which the proportion exposed was the lowest of all five exposure variables (0.212). Since the target group is the unexposed, low proportions of exposed subjects correspond to high proportions of subjects in the target group, and since the groups being compared are the entire sample vs. the target group, it makes sense that high proportions in the target group correspond with low standard errors.

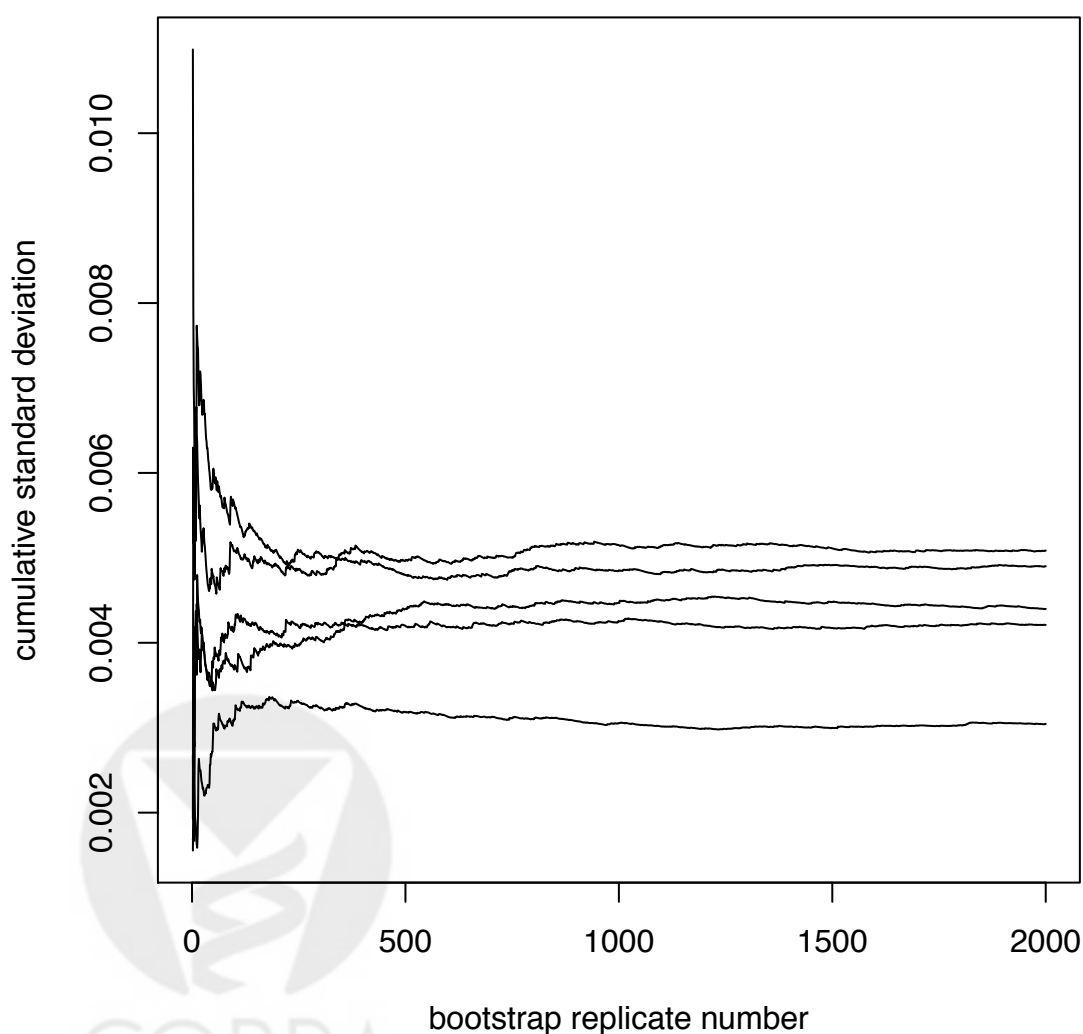
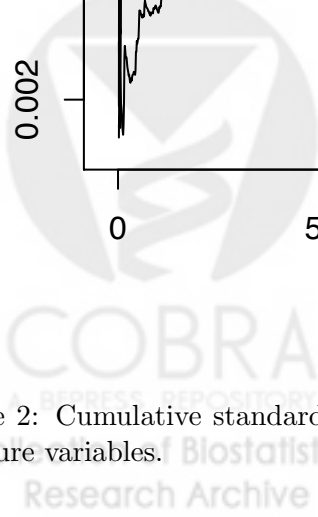


Figure 2: Cumulative standard deviations of the bootstrap parameter estimates for all five exposure variables.



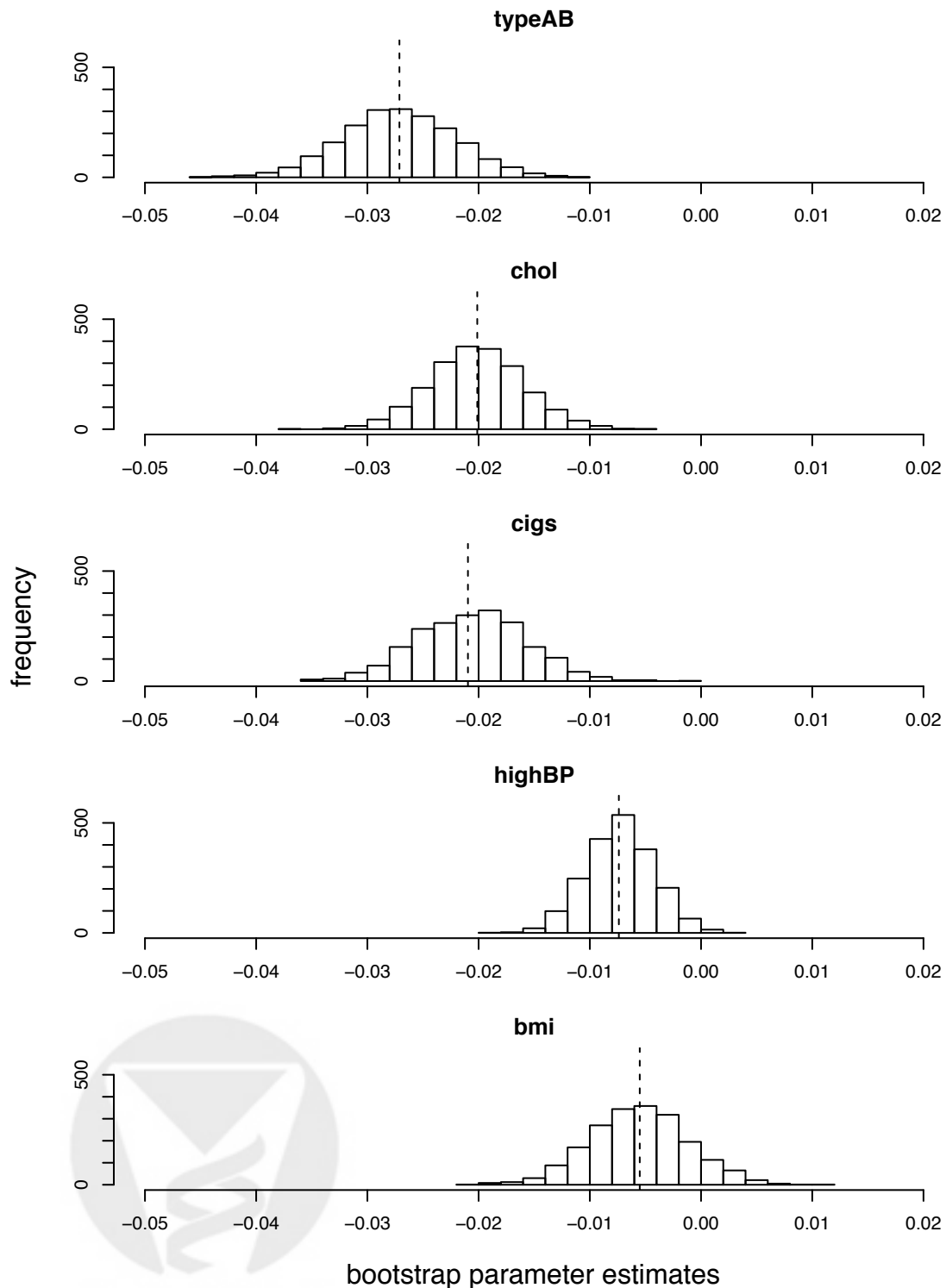


Figure 3: Histograms showing bootstrap distributions of parameter estimates for each exposure variable. Dashed vertical lines show actual parameter estimates. There were 2000 bootstrap replicates.

It is also interesting to note that the maximum off-diagonal value in the correlation matrix (Table 2), which was 0.17, corresponds to the correlation between the variables `bmi` and `highBP`. These are the two variables for which the results were insignificant after Bonferroni adjustment. It is possible that the effect measurements for each of these two variables may have been diluted since the other of the two was also included in the adjustment set. Also, it seems promising that the sum of the (adjusted) parameter estimates (0.0811) is approximately equal to the overall disease rate of 8.2%.

Figure 2 and Figure 3 provide some evidence for the validity of the bootstrap variance estimation procedure. The cumulative standard errors plotted in Figure 2 all appear to converge well before the 2000th bootstrap replicate, and the histograms in Figure 3 indicate that there were no glaring irregularities in the bootstrap distributions of the parameter estimates.

6. Study on water contact and schistosomiasis infection

In this section we provide more example code, using data which was collected as part of the study by Spear *et al.* (2004), and revisited by Sudat *et al.* (2010). This data has also been included as part of the `multiPIM` package and can be loaded with

```
R> data("schisto")
```

6.1. Study background

The study was conducted in a rural area in Sichuan Province, China, to which schistosomiasis is endemic. In November, villagers from 20 villages surrounding Qionghai Lake were interviewed about their activities over the past 7 months that brought them into contact with water. At the end of the infection season, stool sampling and analysis were carried out to determine which of the villagers had become infected.

6.2. Description of data

The `schisto` data frame contains the following columns:

- Outcome variable
 - `stoolpos`: 1 indicates infected, 0 indicates uninfected
- Exposure variables: these record the amount of water exposure of each subject with respect to doing the activities listed, during the 7 month period from April through October
 - `laundwc`: washing clothes or vegetables
 - `toolwc`: washing agricultural tools
 - `bathewc`: washing hands and feet
 - `swimwc`: playing or swimming
 - `ditchwc`: irrigation ditch cleaning and water diverting

- **ricepwc**: planting rice
 - **fishwc**: fishing
- Covariates
 - **village**: a label for the village
 - **age.category**: 5 different age categories (< 18, 18-29, 30-39, 40-49, 50+)

We prepare the *W*, *A* and *Y* data frames:

```
R> W <- data.frame(lapply(schisto[, 9:10], as.factor))
R> A <- schisto[, 2:8]
```

Y will have only one column, so we need to use `drop = FALSE` to prevent it from being turned into a vector:

```
R> Y <- schisto[, 1, drop = FALSE]
```

6.3. The analysis

If one wishes to keep the information contained in continuous exposure variables when using them in the adjustment set, `multiPIM` can be called inside a for loop, with the other columns of *A* included as part of *W*.

To cut down on running time, we use the IPCW estimator and the `rpart` candidate as a stand-alone for `g.method`. This analysis is similar to that performed by [Sudat *et al.* \(2010\)](#) and gives similar results.

```

R> set.seed(23)
R> num.types <- ncol(A)
R> multiPIM.results <- vector("list", length = num.types)
R> for(i in 1:num.types) {
+   A.current <- A[, i, drop = FALSE]
+   A.current[A.current > 0] <- 1
+   W.current <- cbind(W, A[, -i])
+   multiPIM.results[[i]] <- multiPIM(Y, A.current, W.current,
+                                     estimator = "IPCW",
+                                     g.method = "rpart.bin")
+ }
R> results.tab <-
+   t(sapply(multiPIM.results,
+           function(x) summary(x, bf.multiplier = 7)$sum[1,1,]))
R> rownames(results.tab) <- names(A)
R> library("xtable")
R> print(xtable(results.tab, digits = 3), floating = FALSE)

```

| | param.estimate | stand.error | test.stat | p.val | p.val.bon.adj |
|---------|----------------|-------------|-----------|-------|---------------|
| laundwc | -0.010 | 0.013 | 0.769 | 0.442 | 1.000 |
| toolwc | -0.035 | 0.012 | 2.808 | 0.005 | 0.035 |
| bathewc | 0.027 | 0.039 | 0.702 | 0.483 | 1.000 |
| swimwc | -0.013 | 0.017 | 0.754 | 0.451 | 1.000 |
| ditchwc | -0.019 | 0.022 | 0.850 | 0.395 | 1.000 |
| ricepwc | -0.086 | 0.031 | 2.746 | 0.006 | 0.042 |
| fishwc | 0.001 | 0.002 | 0.314 | 0.754 | 1.000 |

Table 4: Results for schisto analysis. The tool washing and rice planting variables remain significant after Bonferroni adjustment.

7. Discussion

The issue of how data can be used to select models and how this should be reflected in the final inference is extremely important in epidemiology today. Ioannidis (2005), in discussing “why most published research findings are false”, writes about the detrimental effects of “flexibility in designs, definitions, outcomes and analytical modes” (page 0698). Leaving the analyst to make arbitrary choices in the model selection process can lead to bias and distorted inference. However, while it is desirable to reduce the flexibility of the analyst to make arbitrary decisions based on the data, it is helpful to make use of automated machine learning methods which are flexible in that they search a large space of possible models.

One common complaint against the complex models which often result from machine learning methods is that they lack interpretability. The solution is the use of low dimensional parameters of interest that have meaningful public health interpretation, and are not tied directly to the model used for $E[Y|A, W]$. By divorcing ourselves from the need to return directly interpretable parameters as coefficients of a regression model, we open the door to

powerful data-adaptive procedures that give hope of consistently estimating the parameter of interest. The researchers can choose the parameter of the data-generating distribution that best addresses the scientific question.

multiPIM accomplishes what has been a contradictory goal: it uses flexible modeling, but still returns asymptotically normal measures of variable importance with robust inference. An additional benefit, which depends on the validity of the causal assumptions and on how the target levels for the exposure variables are chosen, is that the returned parameter estimates have a relevant public health interpretation (they quantify the effect of a hypothetical intervention). Also, since they are on the scale of the outcome, these estimates are directly comparable across different exposure variables, which will typically be on different scales. Due to the machine learning approach, no arbitrary a priori specification of the relevant models is required, and a meaningful ranking of relative importance is provided, with unbiased inference.

Though this is an important first step, there are several ways in which **multiPIM** could be improved, such as 1) by targeting the model selection of the treatment mechanism using a collaborative targeted maximum likelihood estimation approach, 2) by estimating the attributable risk as a smooth function of $A = a$, so that one will not need to dichotomize at a target level, and 3) by providing better integration with the **SuperLearner** and **caret** packages, in order to benefit as much as possible from R's latest machine learning capabilities. However, **multiPIM** represents a new way for researchers to harness the power of machine learning, avoid the bias of arbitrary parametric models, and still get an interpretable measure of relative variable importance, without the typical pitfalls (highlighted in Ioannidis 2005), that plague current exploratory approaches.

Acknowledgments

This research was supported by NIEHS grant R01ES015493-01.

References

- Breiman L (1984). *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA. ISBN 0534980538.
- Breiman L (1996). "Stacked Regressions." *Machine Learning*, **24**(1), 49–64.
- Efron B (1979). "Bootstrap Methods: Another Look at the Jackknife." *The Annals of Statistics*, **7**(1), 1–26.
- Efron B, Hastie T, Johnstone I, Tibshirani R (2004). "Least Angle Regression." *The Annals of Statistics*, **32**(2), pp. 407–451.
- Goeman JJ (2010). "L1 Penalized Estimation in the Cox Proportional Hazards Model." *Biometrical Journal*, **52**(1), 70–84.
- Goeman JJ (2011). *penalized: L1 (Lasso) and L2 (Ridge) Penalized Estimation in GLMs and in the Cox Model*. R package version 0.9-37, URL <http://www.msbi.nl/goeman>.

- Gordis L (2004). *Epidemiology*. 3rd edition. Elsevier.
- Greenland S, Drescher K (1993). “Maximum Likelihood Estimation of the Attributable Fraction from Logistic Models.” *Biometrics*, **49**(3), 865–72.
- Gruber S, van der Laan MJ (2011). “C-TMLE of an Additive Point Treatment Effect.” In [van der Laan and Rose \(2011\)](#), chapter 19.
- Gruber S, van der Laan MJ (2012). “**tmle**: An R Package for Targeted Maximum Likelihood Estimation.” *Journal of Statistical Software*, **51**(13). URL <http://www.jstatsoft.org/v51/i13>.
- Hastie T, Efron B (2011). *lars: Least Angle Regression, Lasso and Forward Stagewise*. R package version 0.9-8, URL CRAN.R-project.org/package=lars.
- Hubbard AE, van der Laan MJ (2008). “Population Intervention Models in Causal Inference.” *Biometrika*, **95**(1), 35–47.
- Ioannidis JPA (2005). “Why Most Published Research Findings Are False.” *PLoS Medicine*, **2**(8), e124. doi:10.1371/journal.pmed.0020124.
- Kooperberg C (2010). *polyspline: Polynomial Spline Routines*. R package version 1.1.5, URL <http://CRAN.R-project.org/package=polyspline>.
- Kooperberg C, Bose S, Stone CJ (1997). “Polychotomous Regression.” *Journal of the American Statistical Association*, **92**(437), 117–127.
- Kuhn M (2008). “Building Predictive Models in R Using the **caret** Package.” *Journal of Statistical Software*, **28**(5), 1–26.
- Kuhn M, Wing J, Weston S, Williams A, Keefer C, Engelhardt A (2011). *caret: Classification and Regression Training*. R package version 5.07-001, URL <http://CRAN.R-project.org/package=caret>.
- L’Ecuyer P, Simard R, Chen EJ, Kelton WD (2002). “An Object-Oriented Random-Number Package with Many Long Streams and Substreams.” *Operations Research*, **50**(6), 1073–1075. doi:10.1287/opre.50.6.1073.358.
- Liaw A, Wiener M (2002). “Classification and Regression by **randomForest**.” *R News*, **2**(3), 18–22. URL <http://CRAN.R-project.org/doc/Rnews/>.
- Messer LC, Oakes JM, Mason S (2010). “Effects of Socioeconomic and Racial Residential Segregation on Preterm Birth: A Cautionary Tale of Structural Confounding.” *American Journal of Epidemiology*, **171**(6), 664–73. doi:10.1093/aje/kwp435.
- Molinaro AM, Simon R, Pfeiffer RM (2005). “Prediction Error Estimation: A Comparison of Resampling Methods.” *Bioinformatics*, **21**(15), 3301–3307.
- Neugebauer R, Bullard J (2010). *DSA: Deletion/Substitution/Addition Algorithm*. R package version 3.1.4, URL <http://www.stat.berkeley.edu/~laan/Software/>.
- Pearl J (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, U.K. ISBN 9780521895606.

- Petersen ML, Porter KE, Gruber S, Wang Y, van der Laan MJ (2011). “Positivity.” In [van der Laan and Rose \(2011\)](#), chapter 10.
- Polley EC, Rose S, van der Laan MJ (2011). “Super Learning.” In [van der Laan and Rose \(2011\)](#), chapter 3.
- Polley EC, van der Laan MJ (2011). *SuperLearner: Super Learner Prediction*. R package version 2.0-4, URL <http://CRAN.R-project.org/package=SuperLearner>.
- R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org/>.
- Robins JM (1986). “A New Approach to Causal Inference in Mortality Studies with a Sustained Exposure Period—Application to Control of the Healthy Worker Survivor Effect.” *Mathematical Modelling*, **7**(9-12), 1393–1512.
- Robins JM (2000). “Marginal Structural Models versus Structural Nested Models as Tools for Causal Inference.” In ME Halloran, D Berry (eds.), *Statistical Models in Epidemiology, the Environment, and Clinical Trials*, pp. 95–113. Springer-Verlag, New York. ISBN 0387989242.
- Rose S, van der Laan MJ (2011a). “Understanding TMLE.” In [van der Laan and Rose \(2011\)](#), chapter 5.
- Rose S, van der Laan MJ (2011b). “Why TMLE?” In [van der Laan and Rose \(2011\)](#), chapter 6.
- Rosenman RH, Brand RJ, Jenkins CD, Friedman M, Straus R, Wurm M (1975). “Coronary Heart Disease in the Western Collaborative Group Study. Final Follow-up Experience of 8 1/2 Years.” *JAMA*, **233**(8), 872–7.
- Rosenman RH, Friedman M, Straus R, Wurm M, Jenkins CD, Messinger HB (1966). “Coronary Heart Disease in the Western Collaborative Group Study. A Follow-up Experience of Two Years.” *JAMA*, **195**(2), 86–92.
- Sevcikova H, Rossini T (2009). *rlecuyer: R Interface to RNG with Multiple Streams*. R package version 0.3-1, URL <http://CRAN.R-project.org/package=rlecuyer>.
- Sinisi SE, Polley EC, Petersen ML, Rhee SY, van der Laan MJ (2007). “Super Learning: An Application to the Prediction of HIV-1 Drug Resistance.” *Statistical Applications in Genetics and Molecular Biology*, **6**(1), Article 7. doi:10.2202/1544-6115.1240.
- Sinisi SE, van der Laan MJ (2004). “Deletion/Substitution/Addition Algorithm in Learning with Applications in Genomics.” *Statistical Applications in Genetics and Molecular Biology*, **3**(1), Article 18. doi:10.2202/1544-6115.1069.
- Spear RC, Seto E, Liang S, Birkner M, Hubbard A, Qiu D, Yang C, Zhong B, Xu F, Gu X, Davis GM (2004). “Factors Influencing the Transmission of *Schistosoma Japonicum* in the Mountains of Sichuan Province of China.” *The American Journal of Tropical Medicine and Hygiene*, **70**(1), 48–56.

- Stone C, Hansen M, Kooperberg C, Truong Y (1997). “Polynomial Splines and their Tensor Products in Extended Linear Modeling.” *The Annals of Statistics*, **25**(4), 1371–1425.
- Sudat SEK, Carlton EJ, Seto EYW, Spear RC, Hubbard AE (2010). “Using Variable Importance Measures from Causal Inference to Rank Risk Factors of Schistosomiasis Infection in a Rural Setting in China.” *Epidemiologic Perspectives & Innovations*, **7**(1), 3.
- Therneau TM, Atkinson B, Ripley BD (2010). *rpart: Recursive Partitioning*. R package version 3.1-46, URL <http://CRAN.R-project.org/package=rpart>.
- Urbanek S (2011). *multicore: Parallel Processing of R Code on Machines with Multiple Cores or CPUs*. R package version 0.1-7, URL <http://CRAN.R-project.org/package=multicore>.
- van der Laan MJ (2006). “Statistical Inference for Variable Importance.” *The International Journal of Biostatistics*, **2**(1), Article 2. doi:10.2202/1557-4679.1008.
- van der Laan MJ, Gill RD, Robins JM (2003). “Locally Efficient Estimation in Censored Data Models: Theory and Examples.” *U.C. Berkeley Division of Biostatistics Working Paper Series*, (Working Paper 85). URL <http://biostats.bepress.com/ucbbiostat/paper85/>.
- van der Laan MJ, Gruber S (2010). “Collaborative Double Robust Targeted Maximum Likelihood Estimation.” *The International Journal of Biostatistics*, **6**(1), Article 17. doi:10.2202/1557-4679.1181.
- van der Laan MJ, Polley EC, Hubbard AE (2007). “Super Learner.” *Statistical Applications in Genetics and Molecular Biology*, **6**(1), Article 25. doi:10.2202/1544-6115.1309.
- van der Laan MJ, Robins JM (2003). *Unified Methods for Censored Longitudinal Data and Causality*. Springer-Verlag, New York. ISBN 0387955569.
- van der Laan MJ, Rose S (2011). *Targeted Learning: Causal Inference for Observational and Experimental Data*. Springer-Verlag, New York. ISBN 9781441997814.
- van der Laan MJ, Rubin D (2006). “Targeted Maximum Likelihood Learning.” *The International Journal of Biostatistics*, **2**(2), Article 11. doi:10.2202/1557-4679.1043.
- Young JG (2007). *Statistical Methods for Complicated Current Status and High-Dimensional Data Structures with Applications in Environmental Epidemiology*. Ph.D. thesis, University of California, Berkeley.
- Young JG, Hubbard AE, Eskenazi B, Jewell NP (2009). “A Machine-Learning Algorithm for Estimating and Ranking the Impact of Environmental Risk Factors in Exploratory Epidemiological Studies.” *U.C. Berkeley Division of Biostatistics Working Paper Series*, (Working Paper 250). URL <http://www.bepress.com/ucbbiostat/paper250>.
- Zheng W, van der Laan MJ (2011). “Cross-Validated Targeted Minimum-Loss-Based Estimation.” In [van der Laan and Rose \(2011\)](#), chapter 27.

Affiliation:

Stephan J. Ritter
Division of Biostatistics
School of Public Health
University of California, Berkeley
E-mail: sritter@berkeley.edu
URL: <http://stat.berkeley.edu/users/sritter>

Nicholas P. Jewell
Division of Biostatistics
School of Public Health and
Department of Statistics
University of California, Berkeley
101 Haviland Hall, MC 7358
Berkeley, CA 94720
United States of America
E-mail: jewell@berkeley.edu
URL: http://works.bepress.com/nicholas_jewell

Alan E. Hubbard
Division of Biostatistics
School of Public Health
University of California, Berkeley
101 Haviland Hall, MC 7358
Berkeley, CA 94720
United States of America
E-mail: hubbard@berkeley.edu
URL: <http://ehs.sph.berkeley.edu/hubbard/>

