Toolkit for Weighting and Analysis of Nonequivalent Groups: A tutorial for the `twang` package

Greg Ridgeway, Dan McCaffrey, Andrew Morral *
RAND

August 6, 2010

1 Introduction

While working on a series of program evaluations for which only observational data were available, we developed methods (written up in McCaffrey *et al.*, 2004) and several R scripts and functions that we believe will be of general interest to analysts developing propensity score models, models of attrition or nonresponse, and some other applications. Because we now regularly use these functions, and many of our colleagues at RAND have found them useful, we have collected them as the Toolkit for Weighting and Analysis of Nonequivalent Groups, or the `twang` package in R.

The propensity score is the probability that a particular case would be assigned or exposed to a treatment condition. Rosenbaum & Rubin (1983) showed that knowing the propensity score is sufficient to separate the effect of a treatment on an outcome from confounding factors that influence both treatment assignment and outcomes, provide the necessary conditions hold. The propensity score has the balancing property that given the propensity score the distribution of features for the treatment cases is the same as that for the control cases. While the treatment selection probabilities are generally not known, good estimates of them can be effective at diminishing or eliminating confounds between pretreatment group differences and treatment outcomes in the estimation of treatment effects..

There are now numerous propensity scoring methods in the literature. They differ in how they estimate the propensity score (e.g. logistic regression, CART), the target estimand (e.g. treatment effect on the treated, population treatment effect), and how they utilize the resulting estimated propensity scores (e.g. stratification, matching, weighting, doubly robust estimators). We originally developed the `twang` package with a particular process in mind, generalized boosted regression to estimate the propensity scores and weighting of the comparison cases to estimate a treatment effect on the treated. The main workhorse of `twang` is the `ps()` function that implements this process. However, the framework of the package is flexible enough to allow the user to use propensity score estimates from other methods and implement new `stop.method` objects to assess the usefulness of those estimates for ensuring equivalence (or "balance") in the pretreatment covariate distributions of treatment and control groups. The same set of functions are also useful for other tasks such as non-response weighting, discussed in section 3.

This package aims to compute good estimates of the propensity scores from the data, check their quality by assessing whether or not they have the balancing properties that we expect in theory, and use them in computing treatment effect estimates.

*The twang package and this tutorial were developed under NIDA grants R01 DA017507 and R01 DA015697-03

2 An example to start

If you have not already done so, install twang by typing `install.packages("twang")`. `twang` relies on other R packages, especially `gbm` and `survey`. You may have to run `install.packages()` for these as well if they are not already installed. You will only need to do this step once. In the future running `update.packages()` regularly will ensure that you have the latest versions of the packages, including bug fixes and new features.

To start using `twang`, first load the package. You will have to do this step once for each R session that you run.

```
> library(twang)
```

```
Loaded gbm 1.6-3.1
```

To demonstrate the package we utilize data from Lalonde's National Supported Work Demonstration analysis (Lalonde 1986, Dehejia & Wahba 1999, `http://www.columbia.edu/~rd247/nswdata.html`). This dataset is provided with the `twang` package.

```
> data(lalonde)
```

R can read data from many other sources. The manual "R Data Import/Export," available at `http://cran.r-project.org/doc/manuals/R-data.pdf`, describes that process in detail.

For the `lalonde` dataset, the variable `treat` is the 0/1 treatment indicator, 1 indicates "treatment" by being part of the National Supported Work Demonstration and 0 indicates "comparison" cases drawn from the Current Population Survey. In order to estimate a treatment effect for this demonstration program that is unbiased by pretreatment group differences on other observed covariates, we include these covariates in a propensity score model of treatment assignment: age, education, black, Hispanic, having no degree, married, earnings in 1974 (pretreatment), and earnings in 1975 (pretreatment). Note that we specify no outcome variables at this time. The `ps()` function is the primary method in `twang` for estimating propensity scores. This step is computationally intensive and can take a few minutes.

```
> par(mfrow = c(1, 2))
> ps.lalonde <- ps(treat ~ age + educ + black +
+     hispan + nodegree + married + re74 + re75,
+     data = lalonde, plots = "optimize", stop.method = stop.methods[c("es.stat.mean",
+         "ks.stat.max")], n.trees = 2000, interaction.depth = 2,
+     shrinkage = 0.01, perm.test.iters = 0, verbose = FALSE)
```

The arguments to `ps()` require some discussion. The first argument specifies a formula indicating that `treat` is the 0/1 treatment indicator and that the propensity score model should predict `treat` from the eight covariates listed there separated by "+". The "+" does *not* mean that these variables are being summed *nor* does it mean that the model is linear. This is just R's notation for variables in the model. There is no need to specify interaction terms in the formula. There is also no need – and it can be counterproductive – to create indicator, or "dummy coded," variables to represent categorical covariates, provided the categorical variables are stored as a `factor` or as `ordered` (see `help(factor)` for more details).

The next argument, `data`, indicates the dataset.

The `plots` argument controls the diagnostic plots that the `ps` function can create. They are described in more detail in Section 2.2. For now `plots="none"` skips the plots, but they can be created later using the `plot()` method. If the call to `ps()` includes an argument `pdf.plots=TRUE` then all the plots are written to a pdf file in the current working directory (use `getwd()` to learn

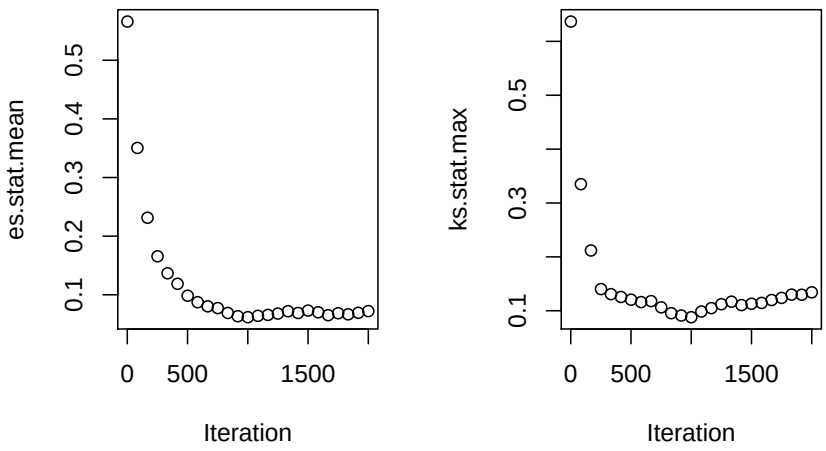


Figure 1: Optimization of `es.stat.mean` and `ks.stat.max`. The horizontal axes indicate the number of gbm iterations and the vertical axes indicate the measure of imbalance between the two groups. For `es.stat.mean` the measure is the average effect size difference across covariates between the treatment and comparison groups and for `ks.stat.max` the measure is the largest of the KS statistics

what your working directory is and `setwd()` to set it). The default is `pdf.plots=FALSE` so that the graphics appear on the screen.

`n.trees`, `interaction.depth`, and `shrinkage` are parameters for the `gbm` model that `ps()` computes and stores. The resulting `gbm` object describes a family of candidate propensity score models indexed by the number of gbm iterations from one to `n.trees`.

The `stop.method` argument takes a `stop.method` object which contains a set of rules and measures for assessing the balance, or equivalence established on the pretreatment covariates of the treatment and weighted control group. The `ps` function selects the optimal number of gbm iterations to minimize the differences between the treatment and control groups as measured by the rules of the given `stop.method` object. Figure 1 illustrates this process. For each panel, the number of gbm iterations is plotted on the horizontal axis and the measure of balance is plotted on the vertical axis. Each iteration adds complexity to the propensity score model giving it greater modeling flexibility. The increased flexibility improves the balance of the two groups up to a certain point after which additional iterations offer no improvement or actually make the balance worse. In this example, iterating for 1031 iterations minimized one measure of balance, the average effect size difference and 972 iterations minimized another measure, the largest of the eight Kolmogorov-Smirnov (KS) statistics computed for the eight covariates. See Section 4.3 for a discussion of these and other balance measures.

The argument `n.trees` is the maximum number of iterations that `ps()` will run; `ps()` will issue a warning if the estimated optimal number of iterations is too close to the bound selected in this argument. Increase `n.trees` or decrease `shrinkage` if this warning appears.

`perm.test.iters` specifies whether p-values for KS statistics should be calculated using Monte Carlo methods, which is slow but can be accurate, or estimated using an analytic approximation that are fast, but produce poor estimates in the presence of many ties. If `perm.test.iters=0`is called, then analytic approximations are used. If `perm.test.iters=500` is called, then 500 Monte Carlo trials are run to establish the reference distribution of KS statistics for each covariate. Higher numbers of trials will produce more precise p-values.

The `gbm` package has various tools for exploring the relationship between the covariates and the treatment assignment indicator if these are of interest. `summary()` computes the relative influence of each variable for estimating the probability of treatment assignment. The gbm estimates depend on the number of iterations at which the gbm model is evaluated, which is specified by the `n.trees` argument in the `summary` method for gbm. In this example, we choose the number of iterations to be the optimal number for minimizing the largest of the KS statistics. This value can be found in the `ps.lalonde$desc$ks.stat.max$n.trees`. Figure 2 shows the barchart of the relative influence if `plot=TRUE` in the call to `summary()`.

```
> summary(ps.lalonde$gbm.obj, n.trees = ps.lalonde$desc$ks.stat.max$n.trees,
+     plot = FALSE)

       var    rel.inf
1    black 47.5272969
2      age 21.8559080
3     re74 15.8316736
4     re75  4.9709651
5     educ  4.5628692
6  married  3.9896479
7 nodegree  0.7613055
8   hispan  0.5003337
```

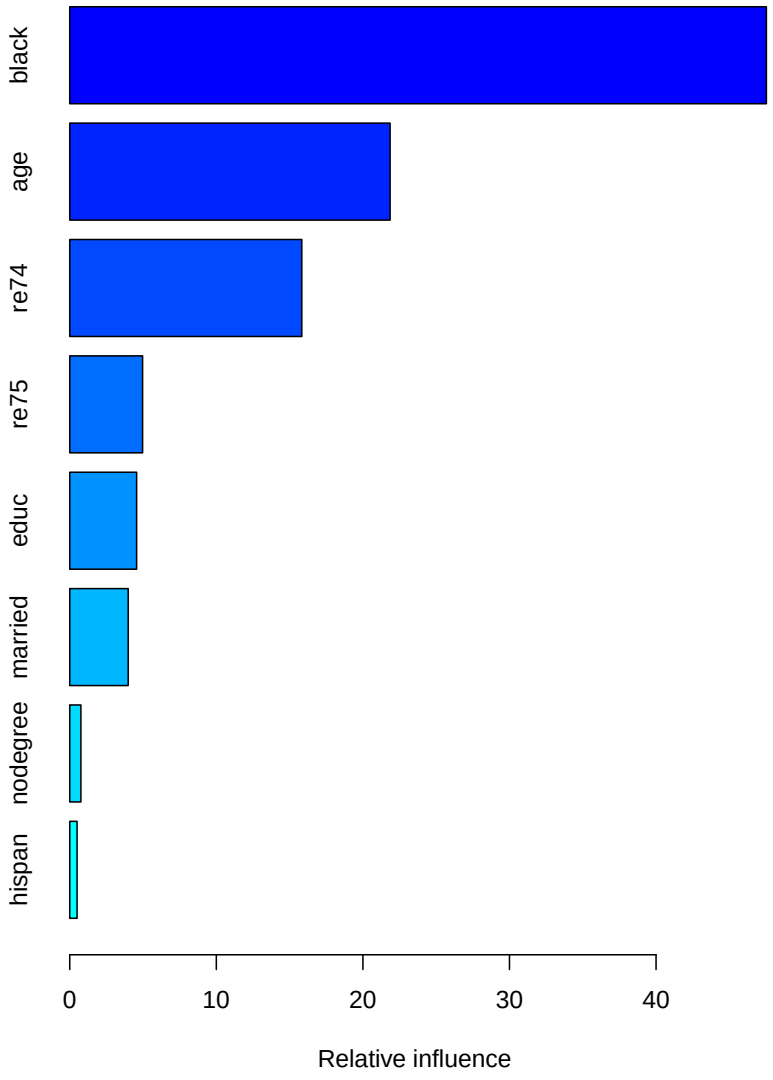


Figure 2: Relative influence of the covariates on the estimated propensity score

2.1 Assessing "balance" using balance tables

Having estimated the propensity scores, `bal.table` produces a table that shows how well the resulting propensity score weights succeed in manipulating the control group so that its weighted pretreatment characteristics match, or balance, those of the unweighted treatment group.

```
> lalonde.balance <- bal.table(ps.lalonde)
> lalonde.balance

$unw
            tx.mn    tx.sd    ct.mn    ct.sd std.eff.sz   stat     p    ks ks.pval
age        25.816    7.155   28.030   10.787     -0.309 -2.994 0.003 0.158   0.003
educ       10.346    2.011   10.235    2.855      0.055  0.547 0.584 0.111   0.074
black       0.843    0.365    0.203    0.403      1.757 19.371 0.000 0.640   0.000
hispan      0.059    0.237    0.142    0.350     -0.349 -3.413 0.001 0.083   0.317
nodegree    0.708    0.456    0.597    0.491      0.244  2.716 0.007 0.111   0.074
married     0.189    0.393    0.513    0.500     -0.824 -8.607 0.000 0.324   0.000
re74     2095.574 4886.620 5619.237 6788.751     -0.721 -7.254 0.000 0.447   0.000
re75     1532.055 3219.251 2466.484 3291.996     -0.290 -3.282 0.001 0.288   0.000

$es.stat.mean
            tx.mn    tx.sd    ct.mn    ct.sd std.eff.sz   stat     p    ks ks.pval
age        25.816    7.155   25.819    7.709      0.000 -0.003 0.998 0.091   0.981
educ       10.346    2.011   10.544    2.204     -0.098 -0.665 0.506 0.083   0.993
black       0.843    0.365    0.845    0.362     -0.006 -0.049 0.961 0.002   1.000
hispan      0.059    0.237    0.046    0.209      0.058  0.647 0.518 0.014   1.000
nodegree    0.708    0.456    0.625    0.485      0.181  0.888 0.375 0.083   0.993
married     0.189    0.393    0.192    0.394     -0.007 -0.054 0.957 0.003   1.000
re74     2095.574 4886.620 1746.830 4145.009      0.071  0.625 0.532 0.057   1.000
re75     1532.055 3219.251 1338.530 2808.704      0.060  0.477 0.634 0.077   0.997

$ks.stat.max
            tx.mn    tx.sd    ct.mn    ct.sd std.eff.sz   stat     p    ks ks.pval
age        25.816    7.155   25.723    7.727      0.013  0.099 0.921 0.088   0.982
educ       10.346    2.011   10.556    2.211     -0.104 -0.712 0.476 0.086   0.984
black       0.843    0.365    0.843    0.364      0.001  0.008 0.993 0.000   1.000
hispan      0.059    0.237    0.046    0.209      0.058  0.656 0.512 0.014   1.000
nodegree    0.708    0.456    0.622    0.486      0.190  0.950 0.343 0.086   0.984
married     0.189    0.393    0.197    0.398     -0.019 -0.139 0.890 0.008   1.000
re74     2095.574 4886.620 1757.163 4156.861      0.069  0.616 0.538 0.056   1.000
re75     1532.055 3219.251 1346.510 2795.059      0.058  0.467 0.641 0.074   0.998
```

`bal.table()` returns information on the pretreatment covariates before and after weighting. The returned object is a list with named components, one for an unweighted analysis (named `unw`) and one for each `stop.method` specified, here `es.stat.mean` and `ks.stat.max`. McCaffrey et al (2004) essentially used `es.stat.mean` for the analyses, but our more recent work has sometimes used `ks.stat.max`. See Section 4.3 for a more detailed description of these choices.

The table contains the following items

tx.mn, ct.mn The treatment means and the propensity score weighted control means for each of the variables. The unweighted table (unw) shows the unweighted means

tx.sd, ct.sd The treatment standard deviations and the propensity score weighted control standard deviations for each of the variables. The unweighted table (unw) shows the unweighted standard deviations

std.eff.sz The standardized effect size, defined as the treatment group mean minus the control group mean divided by the treatment group standard deviation (in discussions of propensity scores this value is sometimes referred to as "standardized bias"). Occasionally, lack of treatment group variance on a covariate results in very large (or infinite) standardized effect sizes. For purposes of analyzing mean effect sizes across multiple covariates, we therefore set all standardized effect sizes larger than 500 to `NA` (missing values).

stat, p Depending on whether the variable is continuous or categorical, `stat` is a t-statistic or a χ^2 statistic. `p` is the associated p-value

ks, ks.pval The Kolmogorov-Smirnov test statistic and its associated p-value. P-values for the KS statistics are either derived from Monte Carlo simulations or analytic approximations, depending on the specifications made in the `perm.test.iters` argument of the `ps` function. For categorical variables this is just the χ^2 test p-value

Components of these tables are useful for demonstrating that pretreatment differences between groups on observed variables have been eliminated using the propensity score weights. The `xtable` package aids in formatting for LaTeXand Word documents. Table 1 shows the results for `ks.stat.max` reformatted for a LaTeXdocument. For Word documents, paste the LaTeXdescription of the table into a Word document, highlight it, Table->Convert->Text to Table, then under "Separate text at" insert "&" in the Other: box. Additional formatting from there will finish it.

```
> library(xtable)
> pretty.tab <- lalonde.balance$ks.stat.max[, c("tx.mn",
+     "ct.mn", "ks")]
> pretty.tab <- cbind(pretty.tab, lalonde.balance$unw[,
+     "ct.mn"])
> names(pretty.tab) <- c("E(Y1|t=1)", "E(Y0|t=1)",
+     "KS", "E(Y0|t=0)")
> xtable(pretty.tab, caption = "Balance of the treatment and comparison groups",
+     label = "tab:balance", digits = c(0, 2, 2,
+         2, 2), align = c("l", "r", "r", "r", "r"))
```

	E(Y1|t=1)	E(Y0|t=1)	KS	E(Y0|t=0)
age	25.82	25.72	0.09	28.03
educ	10.35	10.56	0.09	10.23
black	0.84	0.84	0.00	0.20
hispan	0.06	0.05	0.01	0.14
nodegree	0.71	0.62	0.09	0.60
married	0.19	0.20	0.01	0.51
re74	2095.57	1757.16	0.06	5619.24
re75	1532.06	1346.51	0.07	2466.48

Table 1: Balance of the treatment and comparison groups

The `summary()` method for `ps` objects offers a compact summary of the sample sizes of the groups and the balance measures. If `perm.test.iters>0` was used to create the `ps` object, then

Monte Carlo simulation is used to estimate p-values for the maximum KS statistic that would be expected across the covariates, had individuals with the same covariate values been assigned to groups randomly. Thus, a p-value of 0.04 for `max.ks.p` indicates that the largest KS statistic found across the covariates is larger than would be expected in 96% of trials in which the same cases were randomly assigned to groups.

```
> summary(ps.lalonde)

          type n.treat n.ctrl       ess    max.es
1          unw     185    429 429.00000 1.7567745
2 es.stat.mean     185    429  26.16389 0.1812954
3  ks.stat.max     185    429  28.18081 0.1895653
     mean.es     max.ks max.ks.p    mean.ks iter
1 0.56872589 0.64044604       NA 0.27024507   NA
2 0.06029443 0.09099777       NA 0.05099943 1031
3 0.06398956 0.08798265       NA 0.05153083  972
```

In general, weighted means have greater sampling variance than unweighted means from a sample of equal size. The effective sample size (ESS) of the weighted comparison group captures this increase in variance as

$$\mathrm{ESS} = \frac{\left(\sum_{i \in C} w_i\right)^2}{\sum_{i \in C} w_i^2}. \tag{1}$$

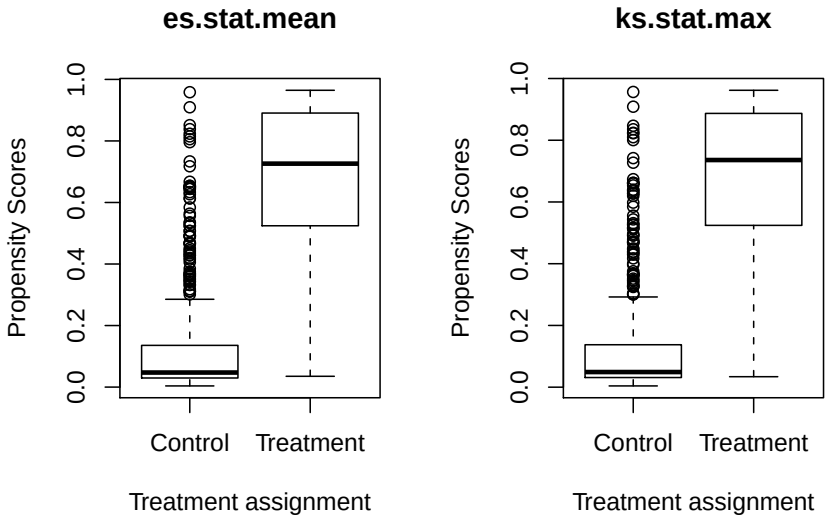
The ESS is approximately the number of observations from a simple random sample needed to obtain an estimate with sampling variation equal to the sampling variation obtained with the weighted comparison observations. Therefore, the ESS will give an estimate of the number of comparison participants that are comparable to the treatment group. The ESS is an accurate measure of the relative size of the variance of means when the weights are fixed or they are uncorrelated with outcomes. Otherwise the ESS underestimates the effective sample size (Little & Vartivarian, 2004). With propensity score weights, it is rare that weights are uncorrelated with outcomes. Hence the ESS might be a lower bound on the effective sample size, but it still serves as a useful measure of the effective number of control cases used in estimating weighted means.

The `ess` column in the summary results shows the ESS for the estimated propensity scores. Note that although the original comparison group had 429 cases, the propensity score estimates effectively utilize only 26.2 or 28.2 of the comparison cases, depending on the rules and measures used to estimate the propensity scores. While this may seem like a large loss of sample size, this indicates that many of the original cases were unlike the treatment cases and, hence, were not useful for isolating the treatment effect. Moreover, similar or even greater reductions in ESS would be expected from alternative approaches to using propensity scores, such as matching or stratification.

2.2 Graphical assessments of balance

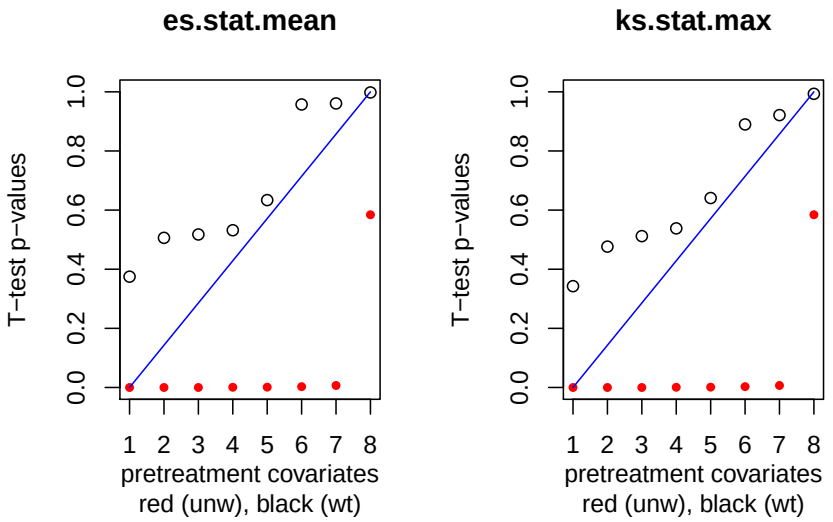
The `plot()` method can generate useful diagnostic plots from the propensity score objects. This command produces boxplots illustrating the spread of the estimated propensity scores in the treatment and comparison groups. Whereas propensity score stratification requires considerable overlap in these spreads, excellent covariate balance can often be achieved with propensity score weights, even when the propensity scores estimated for the treatment and control groups show no overlap.

```
> par(mfrow = c(1, 2))
> plot(ps.lalonde, plots = "ps boxplot")
> par(mfrow = c(1, 1))
```



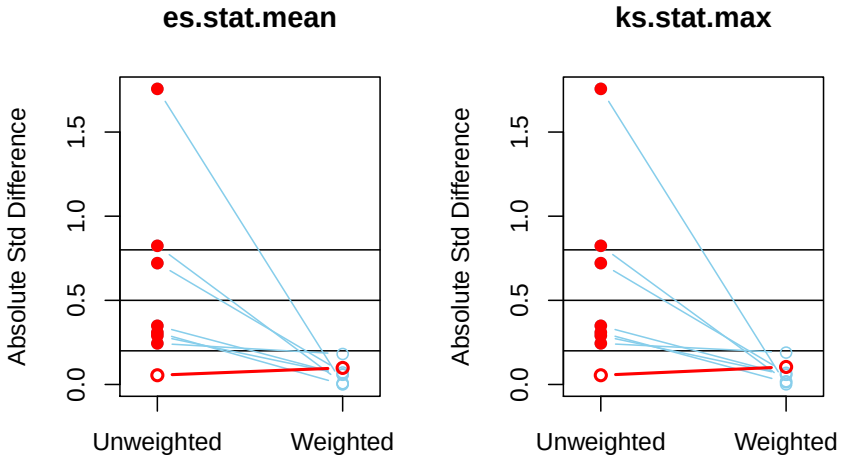
P-values from independent tests in which the null hypothesis is true have a uniform distribution. Therefore, a QQ plot comparing the quantiles of the observed p-values to the quantiles of the uniform distribution illustrate whether group differences observed before and after weighting are consistent with what we would expect to see had groups been formed by random assignment (and hence the null hypothesis would be true). Setting `plots="t pvalues"` generates such QQ plots.

```
> par(mfrow = c(1, 2))
> plot(ps.lalonde, plots = "t pvalues")
> par(mfrow = c(1, 1))
```



Before weighting (closed circles), the groups have statistically significant differences on many variables (i.e., p-values are near zero). After weighting (open circles) the p-values are above the 45-degree line, which represents the cumulative distribution of a uniform variable on [0,1]. This indicates that the p-values are even larger than would be expected in a randomized study. `plot()` can create similar figures for KS statistic p-values by setting `plots="ks pvalues"`.

```
> par(mfrow = c(1, 2))
> plot(ps.lalonde, plots = "es")
> par(mfrow = c(1, 1))
```



The effect size plot illustrates the effect of weights on the magnitude of differences between groups on each pretreatment covariate. These magnitudes are standardized using the standardized effect size described earlier. In these plots, substantial reductions in effect sizes are observed

for most variables (blue lines), with only one variable showing an increase in effect size (red lines), but only a seemingly trivial increase. Closed red circles indicate a statistically significant difference, many of which occur before weighting, none after. In rare cases group differences are very large relative to the treatment group standard deviations. In these cases, a warning appears at the top of the figure indicating that some effect sizes were too large to plot.

2.3 Analysis of outcomes

A separate R package, the `survey` package, is useful for performing the outcomes analyses using propensity score weights. Its statistical methods properly account for the weights when computing standard error estimates. It is not a part of the standard R installation but running `install.packages("survey")` in R will acquire the package from the R archive and install it.

```
> library(survey)
```

The `get.weights` function extracts the propensity score weights from a `ps` object. Those weights may then be used as case weights in a `svydesign` object.

```
> lalonde$w <- get.weights(ps.lalonde, type = "ATT",
+     stop.method = "ks.stat.max")
> design.ps <- svydesign(ids = ~1, weights = ~w,
+     data = lalonde)
```

The `type` argument to the `get.weights` function specifies whether the weights are for estimating the treatment effect on the treated, computed as 1 for the treatment cases and $p/(1-p)$ for the comparison cases, or for estimating the treatment effect on the population, computed as $1/p$ for the treatment cases and $1/(1-p)$ for the comparison cases. The currently implemented `stop.method` objects optimize for the treatment effect on the treated and it is possible that a different set of propensity scores would be optimal for a treatment on the population analysis. The third argument to `get.weights` selects which set of weights to utilize. If no `stop.method` is selected then it returns the first set of weights.

The `svydesign` function from the `survey` package creates an object that stores the dataset along with design information needed for analyses. See `help(svydesign)` for more details on setting up `svydesign` objects.

The aim of the National Supported Work Demonstration analysis is to determine whether the program was effective at increasing earnings in 1978. The propensity score adjusted test can be computed with `svyglm`.

```
> glm1 <- svyglm(re78 ~ treat, design = design.ps)
> summary(glm1)

Call:
svyglm(re78 ~ treat, design = design.ps)

Survey design:
svydesign(ids = ~1, weights = ~w, data = lalonde)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5707.6      756.9   7.541 1.70e-13 ***
treat          641.5      952.0   0.674    0.501
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 49334797)

Number of Fisher Scoring iterations: 2
```

The analysis estimates an increase in earnings of $642 for those that participated in the NSW compared with similarly situated people observed in the CPS. The effect, however, does not appear to be statistically significant.

Some authors have recommended utilizing both propensity score adjustment and additional covariate adjustment to minimize mean square error or to obtain "doubly robust" estimates of the treatment effect (Huppler-Hullsiek & Louis 2002, Bang & Robins 2005). These estimators are consistent if either the propensity scores are estimated correctly *or* the regression model is specified correctly. For example, note that the balance table for `ks.stat.max` made the two groups more similar on `nodegree`, but still some differences remained, 70.8% of the treatment group had no degree while 62.2% of the comparison group had no degree. While linear regression is sensitive to model misspecification when the treatment and comparison groups are dissimilar, the propensity score weighting has made them more similar, perhaps enough so that additional modeling with covariates can adjust for any remaining differences. In addition to potential bias reduction, the inclusion of additional covariates can reduce the standard error of the treatment effect if some of the covariates are strongly related to the outcome.

```
> glm2 <- svyglm(re78 ~ treat + nodegree, design = design.ps)
> summary(glm2)

Call:
svyglm(re78 ~ treat + nodegree, design = design.ps)

Survey design:
svydesign(ids = ~1, weights = ~w, data = lalonde)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6857.0     1248.1   5.494 5.78e-08 ***
treat          801.3      977.8   0.819    0.413
nodegree     -1848.8     1135.5  -1.628    0.104
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 48583321)

Number of Fisher Scoring iterations: 2
```

Adjusting for the remaining group difference in the `nodegree` variable slightly increased the estimate of the program's effect to $801, but the difference is still not statistically significant. We can further adjust for the other covariates, but that too in this case has little effect on the estimated program effect.

```
> glm3 <- svyglm(re78 ~ treat + age + educ + black +
+     hispan + nodegree + married + re74 + re75,
+     design = design.ps)
> summary(glm3)
```

```
Call:
svyglm(re78 ~ treat + age + educ + black + hispan + nodegree +
    married + re74 + re75, design = design.ps)

Survey design:
svydesign(ids = ~1, weights = ~w, data = lalonde)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -1979.3193  4048.3896  -0.489  0.62508
treat         701.4438   931.4407   0.753  0.45170
age             4.3634    51.9186   0.084  0.93305
educ          715.5944   242.7090   2.948  0.00332 **
black        -804.9551   955.8284  -0.842  0.40003
hispan        708.4906  1645.6312   0.431  0.66697
nodegree      485.1592  1499.3160   0.324  0.74636
married       464.5308  1050.1253   0.442  0.65839
re74            0.0418     0.1654   0.253  0.80059
re75            0.1492     0.1746   0.854  0.39332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 46793632)

Number of Fisher Scoring iterations: 2
```

2.4 Estimating the program effect using linear regression

The more traditional regression approach to estimating the program effect would fit a linear model with a treatment indicator and linear terms for each of the covariates.

```
> glm4 <- lm(re78 ~ treat + age + educ + black +
+     hispan + nodegree + married + re74 + re75,
+     data = lalonde)
> summary(glm4)

Call:
lm(formula = re78 ~ treat + age + educ + black + hispan + nodegree +
    married + re74 + re75, data = lalonde)

Residuals:
   Min     1Q Median     3Q    Max
-13595  -4894  -1662   3929  54570

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.651e+01  2.437e+03   0.027   0.9782
treat        1.548e+03  7.813e+02   1.982   0.0480 *
age          1.298e+01  3.249e+01   0.399   0.6897
educ         4.039e+02  1.589e+02   2.542   0.0113 *
black       -1.241e+03  7.688e+02  -1.614   0.1071
```

```
hispan       4.989e+02  9.419e+02   0.530   0.5966
nodegree     2.598e+02  8.474e+02   0.307   0.7593
married      4.066e+02  6.955e+02   0.585   0.5590
re74         2.964e-01  5.827e-02   5.086 4.89e-07 ***
re75         2.315e-01  1.046e-01   2.213   0.0273 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6948 on 604 degrees of freedom
Multiple R-squared: 0.1478,        Adjusted R-squared: 0.1351
F-statistic: 11.64 on 9 and 604 DF,  p-value: < 2.2e-16
```

This model estimates a rather strong treatment effect, estimating a program effect of $1548 with a p-value=0.048. Several variations of this regression approach also estimate strong program effects. For example using square root transforms on the earnings variables yields a p-value=0.016. These estimates, however, are very sensitive to the model structure since the treatment and control subjects differ greatly as seen in the unweighted balance comparison ($unw) from `bal.table(ps.lalonde)`.

2.5 Propensity scores estimated from logistic regression

Propensity score analysis is intended to avoid problems associated with the misspecification of covariate adjusted models of outcomes, but the quality of the balance and the treatment effect estimates can be sensitive to the method used to estimate the propensity scores. Consider estimating the propensity scores using logistic regression instead of `ps()`.

```
> ps.logit <- glm(treat ~ age + educ + black + hispan +
+     nodegree + married + re74 + re75, data = lalonde,
+     family = binomial)
> lalonde$w.logit <- rep(1, nrow(lalonde))
> lalonde$w.logit[lalonde$treat == 0] <- exp(predict(ps.logit,
+     subset(lalonde, treat == 0)))
```

`predict()` for logistic regression model produces estimates on the log-odds scale by default. Exponentiating those predictions for the comparison subjects gives the propensity score weights $p/(1-p)$. `dx.wts()` from the `twang` package diagnoses the balance for an arbitrary set of weights producing a balance table.

```
> bal.logit <- dx.wts(lalonde$w.logit, data = lalonde,
+     vars = c("age", "educ", "black", "hispan",
+         "nodegree", "married", "re74", "re75"),
+     treat.var = "treat", perm.test.iters = 0)



> print(bal.logit)

  type n.treat n.ctrl       ess    max.es    mean.es
1  unw     185    429 429.00000 1.7567745 0.56872589
2          185    429  99.81539 0.1188496 0.03188410
     max.ks    mean.ks iter
1 0.6404460 0.27024507   NA
2 0.3078039 0.09302319   NA
```

For propensity score weights estimated with logistic regression, the largest KS statistic was reduced from the unweighted sample's largest KS of 0.64 to 0.31, which is still quite a large KS statistic. Table 2 shows the details of the balance of the treatment and comparison groups. The means of the two groups appear to be quite similar while the KS statistic shows substantial differences in their distributions.

```
> pretty.tab <- bal.table(bal.logit)[[2]][, c("tx.mn",
+     "ct.mn", "ks")]
> pretty.tab <- cbind(pretty.tab, bal.table(bal.logit)[[1]]$ct.mn)
> names(pretty.tab) <- c("E(Y1|t=1)", "E(Y0|t=1)",
+     "KS", "E(Y0|t=0)")
> xtable(pretty.tab, caption = "Logistic regression estimates of the propensity scores",
+     label = "tab:balancelogit", digits = c(0,
+         2, 2, 2, 2), align = c("l", "r", "r",
+         "r", "r"))
```

	E(Y1|t=1)	E(Y0|t=1)	KS	E(Y0|t=0)
age	25.82	24.97	0.31	28.03
educ	10.35	10.40	0.04	10.23
black	0.84	0.84	0.00	0.20
hispan	0.06	0.06	0.00	0.14
nodegree	0.71	0.69	0.02	0.60
married	0.19	0.17	0.02	0.51
re74	2095.57	2106.05	0.23	5619.24
re75	1532.06	1496.54	0.13	2466.48

Table 2: Logistic regression estimates of the propensity scores

Table 3 compares the balancing quality of the propensity score weights directly with one another.

	n.treat	ess	max.es	mean.es	max.ks	mean.ks
unw	185	429.00	1.76	0.57	0.64	0.27
logit	185	99.82	0.12	0.03	0.31	0.09
es.stat.mean	185	26.16	0.18	0.06	0.09	0.05
ks.stat.max	185	28.18	0.19	0.06	0.09	0.05

Table 3: Summary of the balancing properties of logistic regression and gbm

```
> design.logit <- svydesign(ids = ~1, weights = ~w.logit,
+     data = lalonde)
> glm6 <- svyglm(re78 ~ treat, design = design.logit)
> summary(glm6)

Call:
svyglm(re78 ~ treat, design = design.logit)

Survey design:
```

```
svydesign(ids = ~1, weights = ~w.logit, data = lalonde)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5135.1      588.9   8.719   <2e-16 ***
treat         1214.1      824.7   1.472    0.142
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 49598072)

Number of Fisher Scoring iterations: 2
```

The analysis estimates an increase in earnings of $1214 for those that participated in the NSW compared with similarly situated people observed in the CPS. Table 4 compares all of the treatment effect estimates.

Treatment effect	PS estimate	Linear adjustment
$642	GBM, minimize KS	none
$801	GBM, minimize KS	nodegree
$701	GBM, minimize KS	all
$1548	None	all
$1214	Logistic regression	none
$1237	Logistic regression	all

Table 4: Treatment effect estimates by various methods

3 Non-response weights

The `twang` package was designed to estimate propensity score weights for the evaluation of treatment effects in observational or quasi-experimental studies. However, we find that the package includes functions and diagnostic tools that are highly valuable for other applications, such as for generating and diagnosing nonresponse weights for survey nonresponse or study attrition. We now present an example that uses the tools in `twang`. This example uses the subset of the US Sustaining Effects Study data distributed with the HLM software (Bryk, Raudenbush, Congdon, 1996) and also available in the R package `mlmRev`. The data include mathematics test scores for 1721 students in kindergarten to fourth grade. They also include the students race (Black, Hispanic, or other), gender, an indicator for whether or not the student had been retained in grade, the percent low income students at the school, the school size, the percent of mobile students, the students' grade-levels, student and school IDs, and grades converted to year by centering. The study analysis plans to analyze growth in math achievement from grade 1 to grade 4 using only students with complete data. However, the students with complete data differ from other students. To reduce bias that could potentially result from excluding incomplete cases, our analysis plan is to weight complete cases with nonresponse weights.

The goal of nonresponse weighting is to develop weights for the respondents that make them look like the entire sample – both the respondents and nonrespondents. Since, the respondents already look like themselves, the hard part is to figure out how well each respondent represents the nonrespondents. These two functions served by each respondent can be understood as

two components of the nonresponse weight. Nonresponse weights equal the reciprocal of the probability of response and are applied only to respondents. Let p denote the probability of response and and $1/p$ denote the nonresponse weight. Using basic algebra we can rewrite the nonresponse weights:

$$\frac{1}{p} = 1 + \frac{1-p}{p} \tag{2}$$

Written in this way, the nonresponse weight can be viewed as having two components, a component for the respondent (which equals 1) and a component for the nonrespondents ($(1-p)/p$) that the respondent is to represent in the analysis. This second component is a respondent weight designed to make the respondents look like the non-respondents, which is a problem identical to that which the `ps()` function is designed to solve; it finds weights that make the control group look like the treatment group in terms of the distribution of their covariates. Hence if we call the nonrespondents the "treatment" group and respondents the "control" group then `ps()` function can provide estimates of $(1-p)/p$, the second component of the nonresponse weight, and the diagnostic tools in `twang` can be used to diagnose the weights. To obtain the final nonresponse weight we just add 1 to the weights from `ps()`.

Before we can generate nonresponse weights, we need to prepare the data using the following commands.

First we load the data.

```
> data(egsingle)
```

Next we create the patterns of grades for which students have responses

```
> tmp <- sapply(split(egsingle, egsingle$childid),
+     function(x) {
+         paste(as.character(x$grade), collapse = "")
+     })
```

identify students with test scores for every grade from 1 to 4

```
> tmp <- data.frame(childid = names(tmp), gpatt = tmp,
+     resp = as.numeric((1:length(tmp)) %in% grep("1234",
+         as.character(tmp))))
```

and merge this back to create a single data frame

```
> egsingle <- merge(egsingle, tmp)
```

Because nonresponse is a student-level variable rather than a student-by-year-level variable we create one record per student.

```
> egsingle.one <- unique(egsingle[, -c(3:6)])
```

We also create a race variable

```
> egsingle.one$race <- as.factor(race <- ifelse(egsingle.one$black ==
+     1, 1, ifelse(egsingle.one$hispanic == 1, 2,
+     3)))
```

As discussed above, to use `ps()` to estimate nonresponse, we need to let nonrespondents be the treatment group by modeling an indicator of nonresponse rather than an indicator of response. We create this indicator and then we are set to estimate weights.

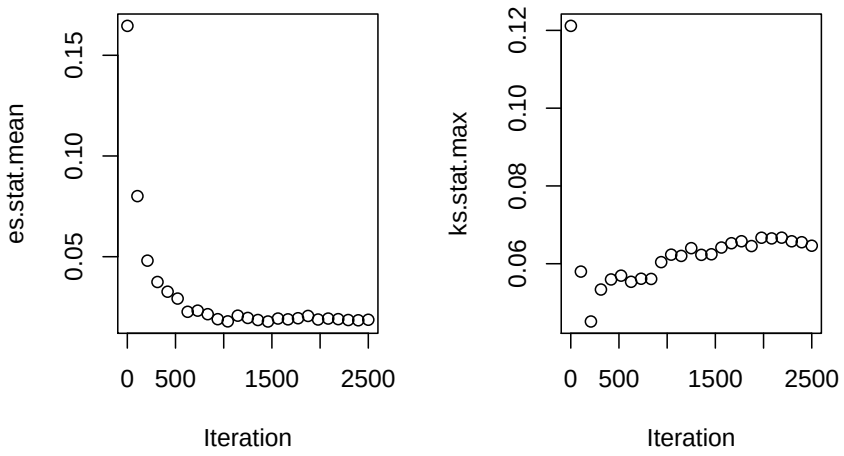


Figure 3: Optimization of `es.stat.mean` and `ks.stat.max` for nonresponse weighting of egsingle data. The horizontal axes indicate the number of iterations and the vertical axes indicate the measure of imbalance between the two groups. For `es.stat.mean` the measure is the average effect size difference between the two groups and for `ks.stat.max` the measure is the largest of the KS statistics

```
> egsingle.one$nresp <- 1 - egsingle.one$resp
```

```
> par(mfrow = c(1, 2))
> egsingle.ps <- ps(nresp ~ race + female + size +
+     lowinc + mobility, data = egsingle.one, plots = "optimize",
+     stop.method = stop.methods[c("es.stat.mean",
+         "ks.stat.max")], n.trees = 2500, verbose = FALSE)
```

The optimal number of iterations for gbm to minimize the maximum KS statistic is 184 and the optimal number of iterations for gbm to minimize the average effect size is 1435. The weights do an excellent job matching the distribution of the respondent group covariates to those of the nonrespondents as shown in Table 5.

```
> pretty.tab <- bal.table(egsingle.ps)$ks.stat.max[,
+     c("tx.mn", "ct.mn", "std.eff.sz", "ks")]
> names(pretty.tab) <- c("Non-responders", "Weighted responders",
+     "Std ES", "KS")
> xtable(pretty.tab, caption = "Balance of the nonrespondents and respondents",
+     label = "tab:balance2", digits = c(0, 2, 2,
+         2, 2), align = c("l", "r", "r", "r", "r"))
```

The final step is to add 1 to the weights to get the final nonresponse weight and then add the nonresponse weights to the respondent data so analyses can proceed.

```
> egsingle.one$wgt <- 1 + get.weights(egsingle.ps,
+     type = "ATT", stop.method = "ks.stat.max")
```

	Non-responders	Weighted responders	Std ES	KS
race:1	0.73	0.71	0.04	0.02
race:2	0.16	0.15	0.04	0.01
race:3	0.11	0.14	-0.10	0.03
female:Female	0.52	0.48	0.08	0.04
female:Male	0.48	0.52	-0.08	0.04
size	761.33	763.58	-0.01	0.04
lowinc	80.75	80.62	0.00	0.04
mobility	36.44	35.40	0.07	0.04

Table 5: Balance of the nonrespondents and respondents

```
> egsinge.resp <- merge(subset(egsingle, subset = resp ==
+     1), subset(egsingle.one, subset = resp ==
+     1, select = c(childid, wgt)))
```

4 The details of `twang`

4.1 Propensity score weighting

Propensity score weighting (Rosenbaum 1987, Wooldridge 2002, Hirano and Imbens 2001, McCaffrey *et al.* 2004) reweights comparison cases so that the distribution of their features match the distribution of features of the treatment cases. Let $f(\mathbf{x}|t=1)$ be the distribution of features for the treatment cases and $f(\mathbf{x}|t=0)$ be the distribution of features for the comparison cases. If treatments were randomized then we would expect these two distributions to be similar. When they differ we will construct a weight, $w(\mathbf{x})$, so that

$$f(\mathbf{x}|t=1) = w(\mathbf{x})f(\mathbf{x}|t=0). \tag{3}$$

For example, if $f(\text{age=}65, \text{sex=F}|t=1) = 0.10$ and $f(\text{age=}65, \text{sex=F}|t=0) = 0.05$ (i.e. 10% of the treatment cases and 5% of the comparison cases are 65 year old females) then we need to give a weight of 2.0 to every 65 year old female in the comparison group so that they have the same representation as in the treatment group. More generally, we can solve (3) for $w(\mathbf{x})$ and apply Bayes Theorem to the numerator and the denominator to give an expression for the propensity score weight for comparison cases,

$$w(\mathbf{x}) = K\frac{f(t=1|\mathbf{x})}{f(t=0|\mathbf{x})} = K\frac{P(t=1|\mathbf{x})}{1-P(t=1|\mathbf{x})}, \tag{4}$$

where K is a normalization constant that will cancel out in the outcomes analysis. Equation (4) indicates that if we assign a weight to comparison case i equal to the odds that a case with features $\mathbf{x}_i$ would be exposed to the treatment, then the distribution of their features would balance. Note that for comparison cases with features that are atypical of treatment cases, the propensity score $P(t=1|\mathbf{x})$ would be near 0 and would produce a weight near 0. On the other hand, comparison cases with features typical of the treatment cases would receive larger weights.

4.2 Estimating the propensity score

In randomized studies $P(t=1|\mathbf{x})$ is known and fixed in the study design. In observational studies the propensity score is unknown and must be estimated, but poor estimation of the

propensity scores can cause just as much of a problem for estimating treatment effects as poor regression modeling of the outcome. Linear logistic regression is the common method for estimating propensity scores, and can suffice for many problems. Linear logistic regression for propensity scores estimates the log-odds of a case being in the treatment given $\mathbf{x}$ as

$$\log \frac{P(t=1|\mathbf{x})}{1-P(t=1|\mathbf{x})} = \beta'\mathbf{x} \tag{5}$$

Usually, β is selected to maximize the logistic log-likelihood

$$\ell\beta = \frac{1}{n}\sum_{i=1}^{n} t_i\beta'\mathbf{x}_i - \log\left(1+\exp(\beta'\mathbf{x}_i)\right) \tag{6}$$

Maximizing (6) provides the maximum likelihood estimates of β. However, in an attempt to remove as much confounding as possible, observational studies often record data on a large number of potential confounders, many of which can be correlated with one another. Standard methods for fitting logistic regression models to such data with the iteratively reweighted least squares algorithm can be statistically and numerically unstable. To improve the propensity score estimates we might also wish to include non-linear effects and interactions in $\mathbf{x}$. The inclusion of such terms only increases the instability of the models.

One increasingly popular method for fitting models with numerous correlated variables is the lasso (least absolute subset selection and shrinkage operator) introduced in statistics in Tibshirani (1996). For logistic regression, lasso estimation replaces (6) with a version that penalizes the absolute magnitude of the coefficients

$$\ell\beta = \frac{1}{n}\sum_{i=1}^{n} t_i\beta'\mathbf{x}_i - \log\left(1+\exp(\beta'\mathbf{x}_i)\right) - \lambda\sum_{j=1}^{J}|\beta_j| \tag{7}$$

The second term on the right-hand side of the equation is the penalty term since it decreases the overall of $\ell\beta$ when there are coefficient that are large in absolute value. Setting $\lambda = 0$ returns the standard (and potentially unstable) logistic regression estimates of β. Setting λ to be very large essentially forces all of the β_j to be equal to 0 (the penalty excludes β_0). For a fixed value of λ the estimated $\hat{\beta}$ can have many coefficients exactly equal to 0, not just extremely small but precisely 0, and only the most powerful predictors of t will be non-zero. As a result the absolute penalty operates as a variable selection penalty. In practice, if we have several predictors of t that are highly correlated with each other, the lasso tends to include all of them in the model, shrink their coefficients toward 0, and produce a predictive model that utilizes all of the information in the covariates, producing a model with greater out-of-sample predictive performance than models fit using variable subset selection methods.

Our aim is to include as covariates all piecewise constant functions of the potential confounders and their interactions. That is, in $\mathbf{x}$ we will include indicator functions for continuous variables like $I(\text{age} < 15), I(\text{age} < 16), \ldots, I(\text{age} < 90)$, etc., for categorical variables like $I(\text{sex} = \text{male}), I(\text{prior MI} = \text{TRUE})$, and interactions among them like $I(\text{age} < 16)I(\text{sex} = \text{male})I(\text{prior MI} = \text{TRUE})$. This collection of basis functions spans a plausible set of propensity score functions, are computationally efficient, and are flat at the extremes of $\mathbf{x}$ reducing the likelihood of propensity score estimates near 0 and 1 that can occur with linear basis functions of $\mathbf{x}$. Theoretically with the lasso we can estimate the model in (7), selecting a λ small enough so that it will eliminate most of the irrelevant terms and yield a sparse model with only the most important main effects and interactions. Boosting (Friedman 2001, 2003, Ridgeway 1999) effectively implements this strategy using a computationally efficient method that Efron *et al.* (2004) showed is equivalent to optimizing (7). With boosting it is possible to maximize (7) for a

range of values of λ with no additional computational effort than for a specific value of λ. We use boosted logistic regression as implemented in the generalized boosted modeling (gbm) package in R (Ridgeway 2005).

4.3 Evaluating the propensity score weights

As with regression analyses, propensity score methods cannot adjust for unmeasured covariates that are uncorrelated with the observed covariates. Nonetheless, the quality of the adjustment for the observed covariates achieved by propensity score weighting is easy to evaluate. The estimated propensity score weights should equalize the distributions of the cases' features as in (3). This implies that weighted statistics of the covariates of the comparison group should equal the same statistics for the treatment group. For example, the weighted average of the age of comparison cases should equal the average age of the treatment cases. To assess the quality of the propensity score weights one could compare a variety of statistics such as means, medians, variances, and Kolmogorov-Smirnov statistics for each covariate as well as interactions. The `twang` package provides both the standardized effect sizes and KS statistics and p-values testing for differences in the means and distributions of the covariates for analysts to use in assessing balance.

The `twang` package encodes decisions on how to assess the quality of the balance in `stop.method` objects which determine how to select the gbm iterations and tune the weights. There are three `stop.method` objects included with `twang` that compare means, KS statistics, and within propensity score strata mean differences. A valid `stop.method` object is a `list` with the following components

metric This is a function that takes weights or propensity scores either as log propensity score weights (useful for direct optimization of the weights), weights for just the control group, or from a model object like `gbm`. This function must also define how missing data are handled and whether to handle each level of a categorical variable separately or to treat them all as one variable. The function must return a single number, smaller values indicating better balance between the two groups. See the help files and code for `ks.stat` and `strat.stat` for examples

rule.summary This is a function that defines how to combine the balance measures across all the variables. `twang` currently uses `mean` and `max`. This function is passed to `metric`

direct This is a logical parameter that indicates whether `twang` should try to directly optimize the weights using a nonlinear optimizer, `nlm`. This method is experimental and can take a very long time for large datasets.

na.action A character string that is passed to metric to indicate how to handle missing data. Current options are

1. "level," treat missing items as a separate level of a categorical variable. If the variable is continuous then separate it out as a distinct variable and try to balance on missingness
2. "exclude," drop missing data
3. "lowest," recode missing data to be the lowest observed value for continuous variables. For factors this is equivalent to "level"

name A character string for labeling results

The `ks.stat.max stop.method` has `metric=ks.stat`, `rule.summary=max`, `direct=FALSE`, `na.action="level"`, and `name="ks.stat.max"`. Advanced users interested in designing and experimenting with their own measures of balance may do so by implementing a new `stop.method` and passing it to the `stop.method` argument of `ps()`.

4.4 Analysis of outcomes

With propensity score analyses the final outcomes analysis is generally straightforward, while the propensity score estimation may require complex modeling. Once we have propensity score weights that equalize the distribution of features of treatment and control cases, we give each treatment case a weight of 1 and each comparison case a weight $w_i = p(\mathbf{x}_i)/(1-p(\mathbf{x}_i))$. We then estimate the treatment effect estimate with a weighted regression model that contains only a treatment indicator. No additional covariates are needed if the propensity score weights account for differences in $\mathbf{x}$.

A combination of propensity score weighting and covariate adjustment can be useful for several reasons. First, the propensity scores may not have been able to completely balance all of the covariates. The inclusion of these covariates in addition to the treatment indicator in a weighted regression model may correct this if the imbalance is relatively small. Second, in addition to exposure, the relationship between some of the covariates and the outcome may also be of interest. Their inclusion can provide coefficients that can estimate the direction and magnitude of the relationship. Third, as with randomized trials, stratifying on covariates that are highly correlated with the outcome can improve the precision of estimates. Lastly, the some treatment effect estimators that utilize an outcomes regression model and propensity scores are "doubly robust" in the sense that if either the propensity score model is correct or the regression model is correct then the treatment effect estimator will be unbiased (Bang & Robins 20005).

References

[1] Bang H. and J. Robins (2005). "Doubly robust estimation in missing data and causal inference models," *Biometrics* 61:692–972.

[2] Dehejia, R.H. and S. Wahba (1999). "Causal effects in nonexperimental studies: re-evaluating the evaluation of training programs," *Journal of the American Statistical Association* 94:1053–1062.

[3] Efron, B., T. Hastie, I. Johnstone, R. Tibshirani (2004). "Least angle regression," *Annals of Statistics* 32(2):407–499.

[4] Friedman, J.H. (2001). "Greedy function approximation: a gradient boosting machine," *Annals of Statistics* 29(5):1189–1232.

[5] Friedman, J.H. (2002). "Stochastic gradient boosting," *Computational Statistics and Data Analysis* 38(4):367–378.

[6] Friedman, J.H., T. Hastie, R. Tibshirani (2000). "Additive logistic regression: a statistical view of boosting," *Annals of Statistics* 28(2):337–374.

[7] Hastie, T., R. Tibshirani, and J. Friedman (2001). *The Elements of Statistical Learning*. Springer-Verlag, New York.

[8] Hirano, K. and G. Imbens (2001). "Estimation of causal effects using propensity score weighting: An application to data on right heart catheterization," *Health Services and Outcomes Research Methodology* 2:259–278.

[9] Huppler-Hullsiek, K. and T. Louis (2002) "Propensity score modeling strategies for the causal analysis of observational data," *Biostatistics* 3:179–193.

[10] Lalonde, R. (1986). "Evaluating the econometric evaluations of training programs with experimental data," *American Economic Review* 76:604–620.

[11] Little, R. J. and S. Vartivarian (2004). "Does weighting for nonresponse increase the variance of survey means?" *ASA Proceedings of the Joint Statistical Meetings*, 3897-3904 American Statistical Association (Alexandria, VA) `http://www.bepress.com/cgi/viewcontent.cgi?article=1034&context=umichbiostat`.

[12] McCaffrey, D., G. Ridgeway, A. Morral (2004). "Propensity score estimation with boosted regression for evaluating adolescent substance abuse treatment," *Psychological Methods* 9(4):403–425.

[13] Ridgeway, G. (1999). "The state of boosting," *Computing Science and Statistics* 31:172–181.

[14] Ridgeway, G. (2005). *GBM 1.5 package manual*. `http://cran.r-project.org/doc/packages/gbm.pdf`.

[15] Ridgeway, G. (2006). "Assessing the effect of race bias in post-traffic stop outcomes using propensity scores." *Journal of Quantitative Criminology* 22(1):1–29.

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

[17] Rosenbaum, P. (1987). "Model-based direct adjustment," *Journal of the American Statistical Association* 82:387–394.

[18] Tibshirani, R. (1996). "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B* 58(1):267–288.

[19] Wooldridge, J. (2002). *Econometric analysis of cross section and panel data*, MIT Press, Cambridge.