Propensity scores for multiple treatments: A tutorial for the **mnps** function in the **twang** package

Lane Burgette, Beth Ann Griffin and Dan McCaffrey [*]
RAND Corporation

January 30, 2015

1   Introduction

The Toolkit for Weighting and Analysis of Nonequivalent Groups, **twang**, was designed to make causal estimates in the binary treatment setting. In **twang** versions 1.3 and later, we have extended this software package to handle more than two treatment conditions through the **mnps** function, which stands for multinomial propensity scores. McCaffrey et al. (2013) describe the methodology behind the **mnps** function; the purpose of this document is to describe the syntax and features related to the implementation in **twang**.

At a high level, the **mnps** function decomposes the propensity score estimation into several applications of the **ps** function, which was designed for the standard dichotomous treatment setting. For this reason, users who are new to **twang** are encouraged to learn about the **ps** function before using the **mnps** function. The other vignette that accompanies the package (Ridgeway et al., 2014) provides an extensive overview of the **ps** function, and much of that information will not be repeated here.

2   An ATE example

To demonstrate the package we use a random subset of the data described in McCaffrey et al. (2013). This truncated dataset is called **AOD**, and is included in the package. There are three treatment groups in the study, and the data include records for 200 youths in each treatment group of an alcohol and other drug treatment evaluation. We begin by loading the package and the data.[1]

```
> library(twang)
> data(AOD)
> set.seed(1)
```

For the **AOD** dataset, the variable **treat** contains the treatment indicators, which have possible values **community**, **metcbt5**, and **scy**. The other variables included in the dataset are:

- **suf12**: outcome variable, substance use frequency at 12 month follow-up

---

[1]Code used in this tutorial can be found in stand alone text file at http://www.rand.org/statistics/twang/downloads.html/mnps_turorial_code.r.

- `illact`: pretreatment covariate, illicit activities scale

- `crimjust`: pretreatment covariate, criminal justice involvement

- `subprob`: pretreatment covariate, substance use problem scale

- `subdep`: pretreatment covariate, substance use dependence scale

- `white`: pretreatment covariate, indicator for non-Hispanic white youth

In such an observational study, there are several quantities that one may be interested in estimating. The estimands that are most commonly of interest are the average treatment effect on the population (ATE) and the average treatment effect on the treated (ATT). The differences between these quantities are explained at length in McCaffrey et al. (2013), but in brief the ATE answers the question of how, on average, the outcome of interest would change if everyone in the population of interest had been assigned to a particular treatment relative to if they had all received another single treatment. The ATT answers the question of how the average outcome would change if everyone who received one particular treatment had instead received another treatment. We first demonstrate the use of `mnps` when ATE is the effect of interest and then turn to using the function to support estimation of ATT.

2.1   Estimating the weights

The main argument for the `mnps` function is a formula with the treatment variable on the left-hand side of a tilde, and pretreatment variables on the right-hand side, separated by plus signs. Other key arguments are `data`, which simply tells the function the name of the dataframe that contains the variables for the propensity score estimation; the `estimand`, which can either be "ATT" or "ATE"; and `verbose`, which if set as `TRUE` instructs the function to print updates on the model fitting process, which can take a few minutes.

```
> mnps.AOD <- mnps(treat ~ illact + crimjust + subprob + subdep + white,
+                  data = AOD,
+                  estimand = "ATE",
+                  verbose = FALSE,
+                  stop.method = c("es.mean", "ks.mean"),
+                  n.trees = 3000)
```

The `twang` methods rely on tree-based regression models that are built in an iterative fashion. As the iterations or number of regression trees added to the model increases, the model becomes more complex. However, at some point, more complex models typically result in worse balance on the pretreatment variables and therefore are less useful in a propensity score weighting context (Burgette, McCaffrey and Griffin, In Press). The `n.trees` argument controls the maximum number of iterations.

Another key choice is the measure of balance that one uses when fitting these models. This is specified in the `stop.method` argument. As with the `ps` function, four `stop.method` objects are included in the package. They are `es.mean`, `es.max`, `ks.mean`, and `ks.max`. The four stopping rules are defined by two components: a balance metric for covariates and rule for summarizing across covariates. The balance metric summarizes the difference between two univariate distributions of a single pretreatment variable (e.g., illicit activities scale). The default stopping rules in `twang` use two balance metrics: absolute standardized mean difference (ASMD; also referred to as the absolute standardized bias or the effect size (ES)) and the Kolmogorov-Smirnov (KS) statistic. The stopping rule use two different rules for summarizing across covariates: the mean of the covariate balance metrics ("mean") or the maximum of the balance metrics ("max"). The

first piece of the stopping rule name identifies the balance metric (ES or KS) and the second piece specifies the method for summarizing across balance metrics. For instance, `es.mean` uses the effect size or ASMD and summarizes across variables with the mean and the `ks.max` uses the KS statistics to assess balances and summarizes using the maximum across variables and the other two stopping rules use the remaining two combinations of balance metrics and summary statistics. In this example, we chose to examine both `es.mean` and `ks.mean`.

After running the `mnps()` command, a useful first step is to make sure that we let the models run for a sufficiently large number of iterations in order to optimize the balance statistics of interest. We do this by seeing whether any of the balance measures of interest still appear to be decreasing after the number of iterations specified by the argument `n.trees` which we set to 3,000 for this example (10,000 iterations is the default).
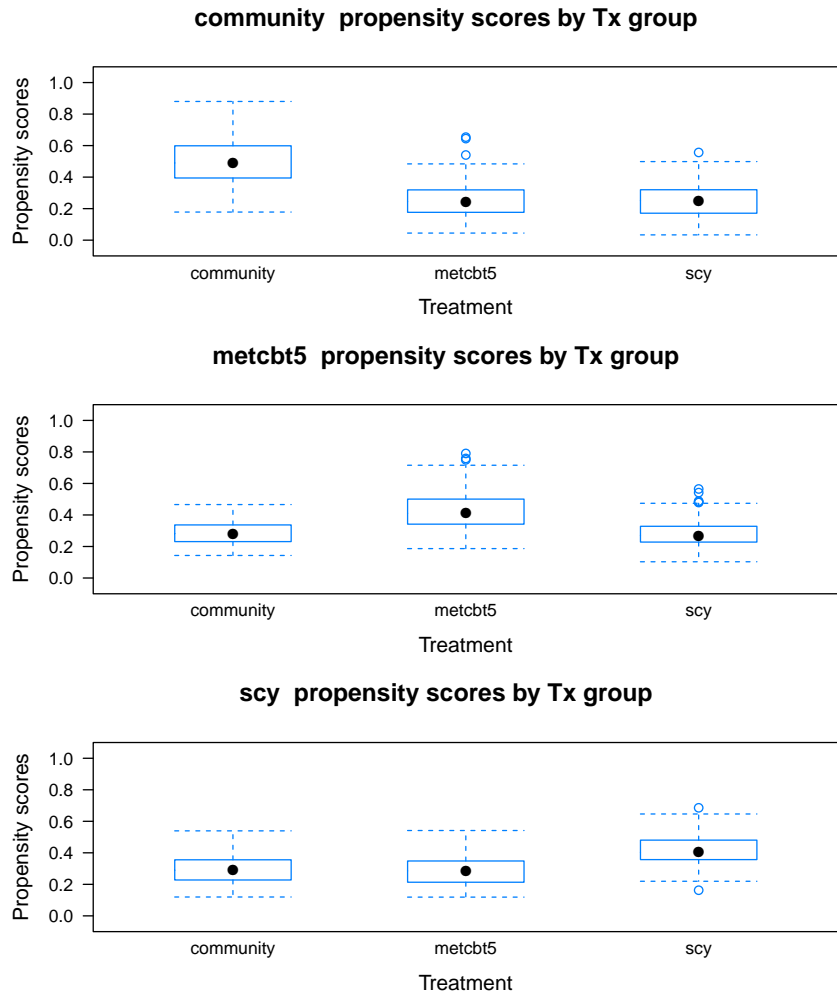
```
>      plot(mnps.AOD, plots = 1)
```
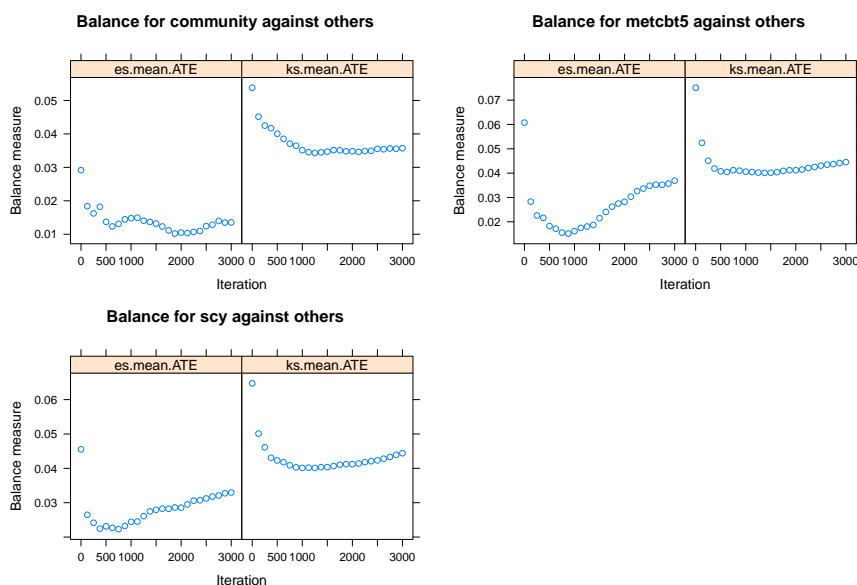

As noted above, `mnps` estimates weights by repeated use of the `ps` function and comparing each treatment the pooled sample of other treatments. The figure has one plot corresponding to each of those fits. Each plot is then further divided into one panel for each stopping rule used in the estimation. Since we used the "es.mean" and "ks.mean" stopping rules there are two panels in each plot. By default the plots for the different treatments are plotted in a single row; setting the height and width of the graphics device can make the plots easier to view. In this figure, it appears that each of the balance measures are optimized with substantially fewer than 3,000 iterations, so we do not have evidence that we should re-run the `mnps()` call with a higher number of iterations or trees.

A key assumption in propensity score analyses is that each experimental unit has a non-zero probability of receiving each treatment. The plausibility of this assumption may be assessed by examining the overlap of the empirical propensity score distributions. This diagnostic is available using the `plots = 2` argument in the `plot` function. We use the `subset` option to specify which stopping rule we wish present in the plot.[2]

---

[2]The value for the `subset` argument can be a character variable with the name of the stopping, as was used in the example code, or a number corresponding to the stopping rule. Stopping rules are numbered by the alphabetical ordering among the rules specified in the `mnps` call.

```
>       plot(mnps.AOD, plots = 2, subset = "es.mean")
```

community propensity scores by Tx group

metcbt5 propensity scores by Tx group

scy propensity scores by Tx group

Here, the overlap assumption generally seems to be met, although there should be some concern that adolescents in the metcbt5 and scy conditions do not overlap well with the community group given the top most graphic. See McCaffrey et al. (2013) for more details on this issue.

2.2   Graphical assessments of balance

As with the `ps` function for the binary treatment setting, the default plotting function for `mnps`-class objects also displays information on commonly-used balance statistics. In particular, when the `plots` argument is set equal to 3, it provides comparisons of the absolute standardized mean differences (ASMD) between the treatment groups on the pretreatment covariates, before and after weighting. When the `plots` argument is set equal to 4, the display is of t- and chi-squared statistic p-values comparing the two groups before and after weighting. However, whereas there is a single plot for these balance diagnostics in the binary treatment setting, in the multiple treatment case, one can either examine a plot for each of the treatment conditions, or summarize the balance statistics in some way, across the treatment conditions. As a default, the

4

`plot` function for an `mnps` object returns the maximum of the pairwise balance statistics across treatment groups for each of the covariates:

```
>       plot(mnps.AOD, plots = 3)
```
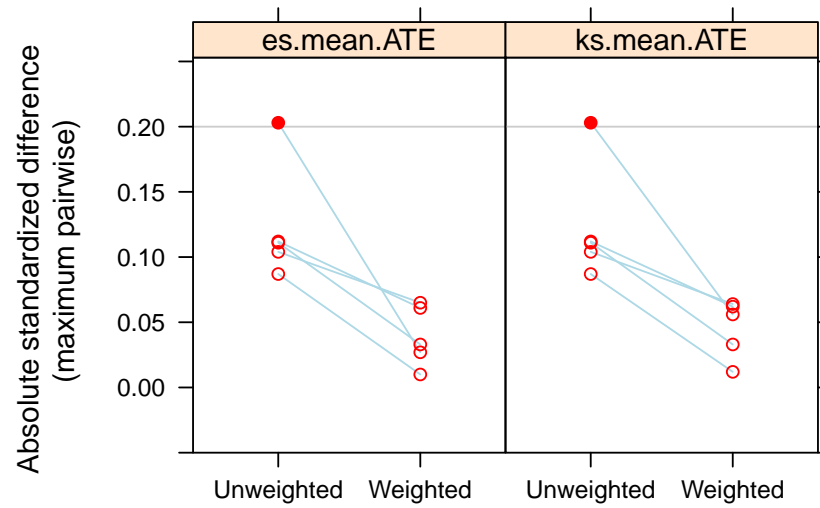

As shown here, after weighting, the maximum ASMD decreases for all pretreatment covariates. The statistically significant difference (before taking the maximum across treatment groups) is indicated by the solid circle. One may see the balance plots for the individual fits by setting the `pairwiseMax` argument to `FALSE`.

```
> plot(mnps.AOD, plots = 3, pairwiseMax = FALSE, figureRows = 3)
```

Balance of community versus metcbt5

Balance of community versus scy

Balance of metcbt5 versus scy
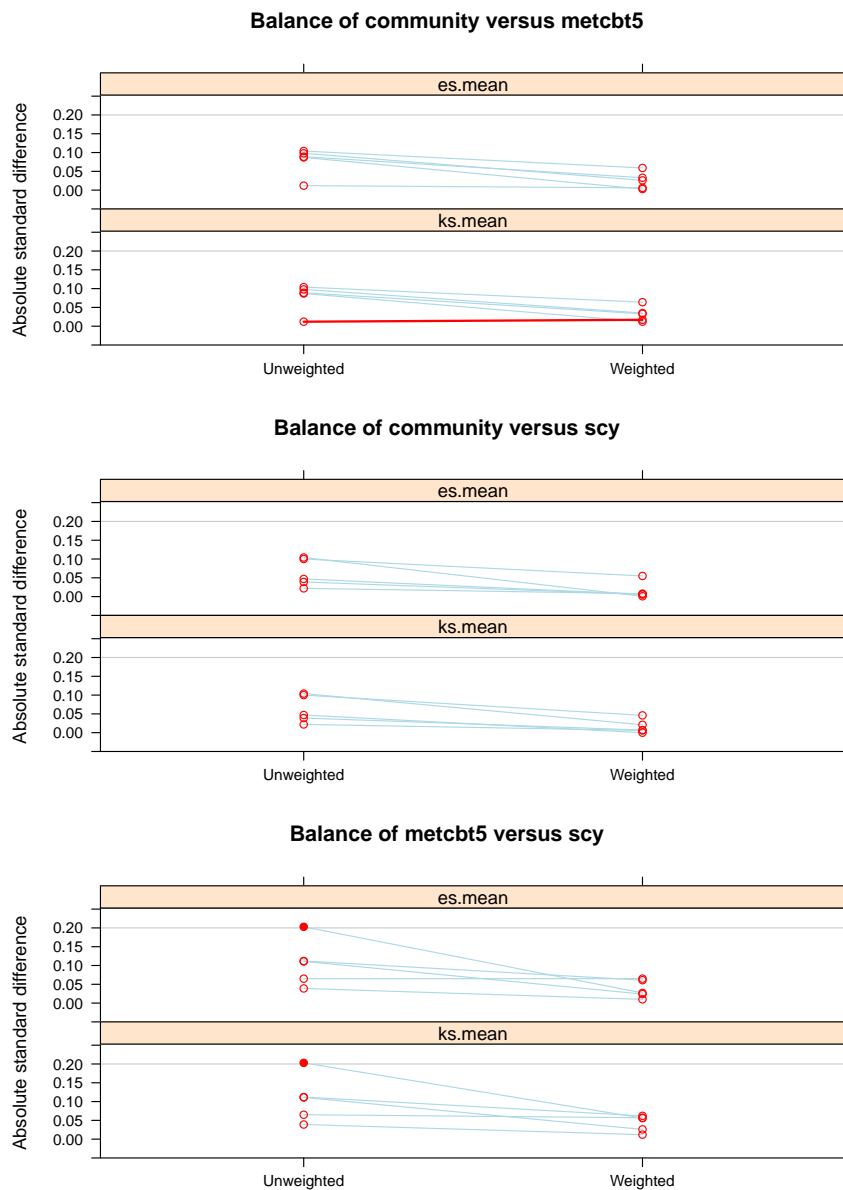
The additional `figureRows` argument instructs the function to spread the plots over three rows; by default the plots would be arranged in a single row rather than a column. We note here that red lines represent pretreatment covariates for which the pairwise ASMDs increase after weighting.

Setting the `plots` argument equal to 4 displays t-test or χ^2 statistic pairwise minimum p-values for differences between each of the individual treatment groups and observations in all other treatment groups.

```
>        plot(mnps.AOD, plots = 4)
```
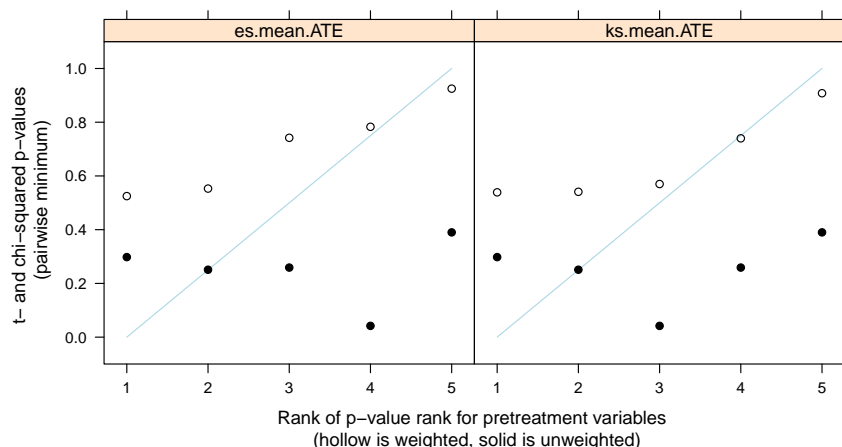

As seen in this figure, the pairwise minimum p-values all increase after propensity score weighting.

Some of the figures include many frames, which can result in figures that are too big or difficult to read for some methods of display. To control this, three controls are available. First, the `treatments` argument can be used to specify only comparisons that involve a specific treatment level or, in the ATE case, only comparisons between two specified treatment levels. Similarly, the `singlePlot` argument . For example, `singlePlot = 2` would display only the second frame of those produced by the plot command (see figure below). Finally, specifying `multiPage = TRUE` prints the frames in succession. If this option is used after specifying a file to plot to (e.g., using `pdf()`), the frames will be printed on separate pages.

```
>        plot(mnps.AOD, plots = 2, subset = "es.mean", singlePlot = 2)
```

metcbt5 propensity scores by Tx group
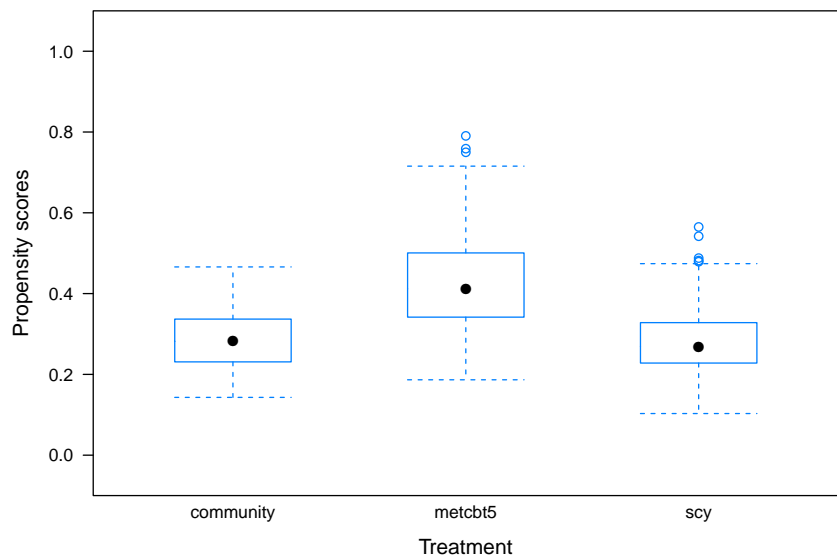
2.3 Tabular assessments of balance

Beyond graphics, there are several other functions that may be of interest to `mnps` users. The first is given by the `bal.table` function. For propensity score analyses with multiple treatments, this function returns a lot of information. The intention with this function is that its output be loaded into a spreadsheet software program. (E.g., one can write the output into a .csv file using the `write.csv` function and open the resulting file using a spreadsheet application.) For each outcome category, and each stopping rule (in addition to the unweighted analysis) the `bal.table` function gives balance statistics such as weighted and unweighted means by treatment group.

```
> bal.table(mnps.AOD, digits = 2)
```

	tmt1	tmt2	var	mean1	mean2	pop.sd	std.eff.sz
1	community	metcbt5	illact	0.10	0.01	1.01	0.09
2	community	metcbt5	crimjust	-0.07	0.04	1.04	0.10
3	community	metcbt5	subprob	-0.06	0.03	0.98	0.09
4	community	metcbt5	subdep	0.05	0.06	1.03	0.01
5	community	metcbt5	white	0.16	0.20	0.38	0.10
6	community	scy	illact	0.10	0.12	1.01	0.02
7	community	scy	crimjust	-0.07	-0.17	1.04	0.10
8	community	scy	subprob	-0.06	-0.01	0.98	0.05
9	community	scy	subdep	0.05	-0.06	1.03	0.10
10	community	scy	white	0.16	0.18	0.38	0.04
11	metcbt5	scy	illact	0.01	0.12	1.01	0.11
12	metcbt5	scy	crimjust	0.04	-0.17	1.04	0.20
13	metcbt5	scy	subprob	0.03	-0.01	0.98	0.04
14	metcbt5	scy	subdep	0.06	-0.06	1.03	0.11
15	metcbt5	scy	white	0.20	0.18	0.38	0.07
16	community	metcbt5	illact	0.09	0.05	1.01	0.03
17	community	metcbt5	crimjust	-0.09	-0.06	1.04	0.03
18	community	metcbt5	subprob	-0.01	-0.02	0.98	0.00
19	community	metcbt5	subdep	0.02	0.02	1.03	0.01
20	community	metcbt5	white	0.17	0.20	0.38	0.06
21	community	scy	illact	0.09	0.08	1.01	0.01
22	community	scy	crimjust	-0.09	-0.09	1.04	0.00
23	community	scy	subprob	-0.01	-0.01	0.98	0.01
24	community	scy	subdep	0.02	-0.04	1.03	0.06
25	community	scy	white	0.17	0.17	0.38	0.01
26	metcbt5	scy	illact	0.05	0.08	1.01	0.02
27	metcbt5	scy	crimjust	-0.06	-0.09	1.04	0.03
28	metcbt5	scy	subprob	-0.02	-0.01	0.98	0.01
29	metcbt5	scy	subdep	0.02	-0.04	1.03	0.06
30	metcbt5	scy	white	0.20	0.17	0.38	0.07
31	community	metcbt5	illact	0.08	0.05	1.01	0.03
32	community	metcbt5	crimjust	-0.08	-0.05	1.04	0.03
33	community	metcbt5	subprob	0.00	-0.01	0.98	0.01
34	community	metcbt5	subdep	0.01	0.02	1.03	0.02
35	community	metcbt5	white	0.17	0.19	0.38	0.06
36	community	scy	illact	0.08	0.08	1.01	0.01
37	community	scy	crimjust	-0.08	-0.11	1.04	0.02
38	community	scy	subprob	0.00	0.00	0.98	0.00

```
39 community    scy   subdep  0.01 -0.04  1.03      0.05
40 community    scy    white  0.17  0.17  0.38      0.01
41   metcbt5    scy   illact  0.05  0.08  1.01      0.03
42   metcbt5    scy crimjust -0.05 -0.11  1.04      0.06
43   metcbt5    scy  subprob -0.01  0.00  0.98      0.01
44   metcbt5    scy   subdep  0.02 -0.04  1.03      0.06
45   metcbt5    scy    white  0.19  0.17  0.38      0.06
       p   ks ks.pval stop.method
1   0.38 0.10    0.27         unw
2   0.33 0.10    0.22         unw
3   0.39 0.09    0.39         unw
4   0.91 0.06    0.92         unw
5   0.30 0.04    1.00         unw
6   0.82 0.06    0.87         unw
7   0.30 0.08    0.55         unw
8   0.63 0.09    0.39         unw
9   0.31 0.08    0.47         unw
10  0.69 0.02    1.00         unw
11  0.26 0.11    0.18         unw
12  0.04 0.13    0.07         unw
13  0.70 0.06    0.79         unw
14  0.25 0.09    0.39         unw
15  0.52 0.02    1.00         unw
16  0.74 0.06    0.90     es.mean
17  0.79 0.05    0.93     es.mean
18  0.97 0.06    0.83     es.mean
19  0.96 0.05    0.96     es.mean
20  0.58 0.02    1.00     es.mean
21  0.94 0.05    0.97     es.mean
22  0.99 0.04    1.00     es.mean
23  0.95 0.07    0.77     es.mean
24  0.58 0.05    0.96     es.mean
25  0.95 0.00    1.00     es.mean
26  0.81 0.06    0.79     es.mean
27  0.78 0.06    0.90     es.mean
28  0.92 0.04    1.00     es.mean
29  0.55 0.06    0.79     es.mean
30  0.53 0.03    1.00     es.mean
31  0.74 0.06    0.84     ks.mean
32  0.72 0.05    0.95     ks.mean
33  0.91 0.05    0.93     ks.mean
34  0.87 0.05    0.97     ks.mean
35  0.54 0.02    1.00     ks.mean
36  0.95 0.05    0.98     ks.mean
37  0.83 0.04    1.00     ks.mean
38  1.00 0.06    0.88     ks.mean
39  0.65 0.05    0.96     ks.mean
40  0.95 0.00    1.00     ks.mean
41  0.80 0.06    0.83     ks.mean
42  0.57 0.06    0.81     ks.mean
```

```
43 0.91 0.04    1.00      ks.mean
44 0.54 0.06    0.80      ks.mean
45 0.58 0.02    1.00      ks.mean
```

As of version 1.4 of TWANG, the balance measures are given for all pairwise combinations. (Prior to that version the balance measures were reported for each treatment against all others; we feel that the pairwise comparisons give a fuller accounting of balance in ATE applications.)

More parsimonious versions of the summaries are available using the `collapse.to` argument. Setting `collapse.to = 'covariate'` gives the maximum of the ASMD and the minimum of the p-value across all pairwise comparisons for each pretreatment covariate and stopping rule.

```
> bal.table(mnps.AOD, collapse.to = 'covariate', digits = 4)

         var max.std.eff.sz  min.p max.ks min.ks.pval
1     illact          0.1112 0.2591 0.1100      0.1779
2   crimjust          0.2027 0.0416 0.1300      0.0680
3    subprob          0.0867 0.3896 0.0900      0.3935
4     subdep          0.1120 0.2514 0.0900      0.3935
5      white          0.1044 0.2984 0.0400      0.9973
6     illact          0.0326 0.7421 0.0647      0.7934
7   crimjust          0.0272 0.7827 0.0568      0.8964
8    subprob          0.0097 0.9248 0.0666      0.7664
9     subdep          0.0605 0.5529 0.0646      0.7944
10     white          0.0653 0.5253 0.0250      1.0000
11    illact          0.0327 0.7401 0.0625      0.8322
12  crimjust          0.0559 0.5705 0.0639      0.8086
13   subprob          0.0116 0.9077 0.0583      0.8795
14    subdep          0.0622 0.5410 0.0646      0.7985
15     white          0.0645 0.5395 0.0247      1.0000
   stop.method
1          unw
2          unw
3          unw
4          unw
5          unw
6      es.mean
7      es.mean
8      es.mean
9      es.mean
10     es.mean
11     ks.mean
12     ks.mean
13     ks.mean
14     ks.mean
15     ks.mean
```

As shown, for each pretreatment variable, the maximum ASMD has decreased and the minimum p-values have increased after applying weights that arise from either stop.method.

Another useful summary table sets `collapse.to = 'stop.method'` which further collapses the results above so that we summarize balance across all covariates and all pairwise group comparisons.

10

```
> bal.table(mnps.AOD, collapse.to = 'stop.method', digits = 4)

  max.std.eff.sz  min.p max.ks min.ks.pval stop.method
1         0.2027 0.0416 0.1300      0.0680         unw
2         0.0653 0.5253 0.0666      0.7664     es.mean
3         0.0645 0.5395 0.0646      0.7985     ks.mean
```

Here we quickly see how the maximum ASMDs and minimum p-values have all moved in the desired direction after propensity score weighting.

Rather than collapsing the values of the table as described above, there are also several options for subsetting the `bal.table` output. The arguments `subset.var` and `subset.stop.method` instruct the function to include only the covariates indicated, and stop.method results indicated, respectively. The `subset.treat` instructs the function to return only the pairwise comparisons including the specified treatment or, if two treatment levels are indicated, the pair-wise comparisons that include those two treatments. Note that `subset.treat` may not be used when `collapse.to` is specified as 'stop.method' or 'covariate'. Further, the table may be subset on the basis of ES and KS and the related p-values via the `es.cutoff`, `ks.cutoff`, `p.cutoff`, and `ks.p.cutoff` arguments. These cutoffs exclude rows that are well-balanced as measured by the corresponding . For example `p.cutoff = 0.1` would exclude rows with p-values greater than 10%, and `es.cutoff = 0.2` excludes rows with ES values below 0.2 in absolute value. Examples of the use of these subsetting arguments are given below.

```
> bal.table(mnps.AOD, subset.treat = c('community', 'metcbt5'),
+           subset.var = c('white', 'illact', 'crimjust'))

         tmt1    tmt2      var  mean1  mean2 pop.sd
1   community metcbt5   illact  0.097  0.007  1.014
2   community metcbt5 crimjust -0.065  0.037  1.041
5   community metcbt5    white  0.160  0.200  0.383
16  community metcbt5   illact  0.085  0.052  1.014
17  community metcbt5 crimjust -0.092 -0.065  1.041
20  community metcbt5    white  0.173  0.195  0.383
31  community metcbt5   illact  0.083  0.050  1.014
32  community metcbt5 crimjust -0.084 -0.048  1.041
35  community metcbt5    white  0.169  0.194  0.383
   std.eff.sz     p    ks ks.pval stop.method
1       0.089 0.385 0.100   0.270         unw
2       0.098 0.328 0.105   0.221         unw
5       0.104 0.298 0.040   0.997         unw
16      0.033 0.742 0.057   0.896     es.mean
17      0.026 0.793 0.054   0.931     es.mean
20      0.059 0.582 0.023   1.000     es.mean
31      0.033 0.740 0.062   0.839     ks.mean
32      0.035 0.723 0.052   0.950     ks.mean
35      0.064 0.539 0.025   1.000     ks.mean

> bal.table(mnps.AOD, subset.stop.method = 'es.mean', collapse.to = 'covariate')

       var max.std.eff.sz min.p max.ks min.ks.pval
6   illact          0.033 0.742  0.065       0.793
7 crimjust          0.027 0.783  0.057       0.896
8  subprob          0.010 0.925  0.067       0.766
```

```
9      subdep          0.061 0.553  0.065       0.794
10      white          0.065 0.525  0.025       1.000
    stop.method
6       es.mean
7       es.mean
8       es.mean
9       es.mean
10      es.mean

> bal.table(mnps.AOD, es.cutoff = 0.1)

           tmt1     tmt2      var  mean1  mean2 pop.sd
5   community metcbt5    white  0.160  0.200  0.383
7   community      scy crimjust -0.065 -0.174  1.041
9   community      scy   subdep  0.046 -0.058  1.031
11    metcbt5      scy   illact  0.007  0.120  1.014
12    metcbt5      scy crimjust  0.037 -0.174  1.041
14    metcbt5      scy   subdep  0.058 -0.058  1.031
   std.eff.sz     p   ks ks.pval stop.method
5       0.104 0.298 0.040   0.997         unw
7       0.104 0.295 0.080   0.545         unw
9       0.100 0.312 0.085   0.466         unw
11      0.111 0.259 0.110   0.178         unw
12      0.203 0.042 0.130   0.068         unw
14      0.112 0.251 0.090   0.394         unw
```

Finally, there is also `summary` method for the `mnps` objects which gives the collapsed version of `bal.table()` as well as information about the effective sample sizes for each treatment group under each stop.method. The `summary` function for an mnps output object does not have a `digits` argument.

```
> summary(mnps.AOD)

Summary of pairwise comparisons:
  max.std.eff.sz       min.p     max.ks min.ks.pval
1     0.20266446 0.04161562 0.13000000   0.0680192
2     0.06529298 0.52525235 0.06661985   0.7663959
3     0.06448455 0.53947426 0.06459093   0.7985293
  stop.method
1        unw
2     es.mean
3     ks.mean


Sample sizes and effective sample sizes:
  treatment    n ESS.es.mean ESS:ks.mean
1 community 200    184.5124    187.4713
2   metcbt5 200    186.1874    183.3987
3       scy 200    189.5017    185.7009
```

After examining the graphical and tabular diagnostics provided by `twang`, we can analyze the outcome variable using the propensity scores generated by the `mnps` function. Although two stop methods were specified initially (`es.mean` and `ks.mean`), at this point we have to commit

to a single set of weights. From the `bal.table` call above, we see that the balance properties are very similar for the two stopping rules, and from the `summary` statement, we see that the effective sample sizes (ESS) are similar as well. Hence, we expect the two stop methods to give similar results; we choose to analyze the data with the `es.mean` weights.

2.4 Estimating treatment effects

In order to analyze the data using the weights, it is recommended that one use the `survey` package, which performs weighted analyses. We can add the weights to the dataset using the `get.weights` function and specify the survey design as follows:

```
> library(survey)
> AOD$w <- get.weights(mnps.AOD, stop.method = "es.mean")
> design.mnps <- svydesign(ids=~1, weights=~w, data=AOD)
```

As shown in the `ps` vignette, we can then perform the propensity score-adjusted regression using the `svyglm` function:

```
> glm1 <- svyglm(suf12 ~ as.factor(treat), design = design.mnps)
> summary(glm1)

Call:
svyglm(formula = suf12 ~ as.factor(treat), design = design.mnps)

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

Coefficients:
                        Estimate Std. Error t value
(Intercept)             -0.09913    0.06736  -1.472
as.factor(treat)metcbt5  0.14858    0.10502   1.415
as.factor(treat)scy      0.06464    0.09998   0.647
                        Pr(>|t|)
(Intercept)                0.142
as.factor(treat)metcbt5    0.158
as.factor(treat)scy        0.518

(Dispersion parameter for gaussian family taken to be 1.002082)

Number of Fisher Scoring iterations: 2
```

By default, `svyglm` includes dummy variables for MET/CBT-5 and SCY, Community is the holdout group (the holdout is the group with the label that comes first alphabetically). Consequently, the estimated effect for MET/CBT-5 equals the weighted mean for the MET/CBT-5 sample less the weighted mean for the Community sample, where both means are weighted to match the overall sample. Similarly, the effect fro SCY equals the difference in the weighted means for the SCY and Community samples. The coefficients estimate the causal effects of MET/CBT-5 vs. Community and SCY vs. Community, respectively, assuming there are no unobserved confounders. Using this small subset of the data, we are unable to detect differences in the treatment group means. In the context of this application, the signs of the estimates correspond to higher substance use frequency for youths exposed to MET/CBT-5 or SCY relative

to Community. More details on how to obtain all relevant pairwise differences can be found in McCaffrey et al. (2013).

As an alternative to estimating the pairwise differences, we could also estimate the causal effect of each treatment relative to the average potential outcome of all the treatments. This estimate is easy to obtain using `svyglm` through the use of the `constrast` argument in the function.

```
> glm2 <- svyglm(suf12 ~ treat, design = design.mnps, contrast=list(treat=contr.sum))
> summary(glm2)

Call:
svyglm(formula = suf12 ~ treat, design = design.mnps, contrast = list(treat = contr.sum))

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

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.02805    0.04280  -0.655    0.512
treat1      -0.07108    0.05783  -1.229    0.220
treat2       0.07751    0.06322   1.226    0.221

(Dispersion parameter for gaussian family taken to be 1.002082)

Number of Fisher Scoring iterations: 2
```

The function now provides the estimates for Community and MET/CBT-5. It labels them "treat1" and "treat2" because it uses their numeric codings rather than the factor levels. We have seen previously that the factor levels for treatment are "community", "metcbt5", and "scy" as levels, 1, 2, and 3. Relative to the average of all the treatments, the weighted Community group has lower substance use and the weighted MET/CBT-5 group has higher use. The SCY estimate is not reported because it is a linear combination of the other to estimates. It can be found by:

```
> -sum(coef(glm2)[-1])

[1] -0.006432045
```

The standard error of this estimate can be calculated using the covariance matrix for the estimated coefficients:

```
> sqrt(c(-1,-1) %*% summary(glm2)$cov.scaled[-1,-1] %*% c(-1,-1))

          [,1]
[1,] 0.0604305
```

The SCY mean is about equal to the average and the difference between them is very small relative to its standard error.

3 An ATT example

3.1 Estimating the weights

It is also possible to explore treatment effects on the treated (ATTs) using the `mnps` function. A key difference in the multiple treatment setting is that we must be clear as to which treatment

condition "the treated" refers to. This is done through the `treatATT` argument. Here, we define the treatment group of interest to be the community group; thus, we are trying to draw inferences about the relative effectiveness of the three treatment groups for individuals like those who were enrolled in the community program.

```
> mnps.AOD.ATT <- mnps(treat ~ illact + crimjust + subprob + subdep + white,
+                      data = AOD,
+                      estimand = "ATT",
+                      treatATT = "community",
+                      verbose = FALSE,
+                      n.trees = 3000,
+                      stop.method = c("es.mean", "ks.mean"))
```
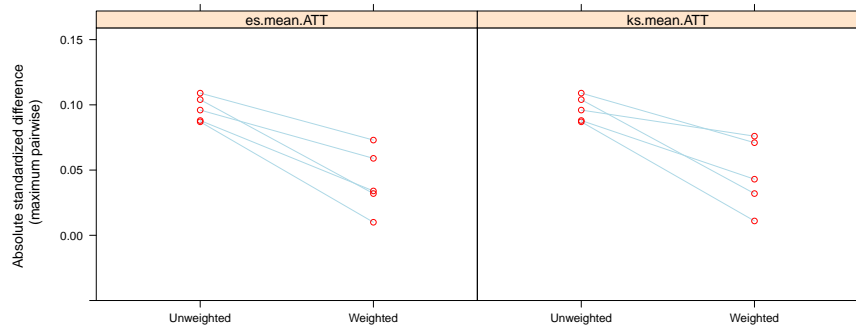
3.2 Graphical assessments of balance

The same basic graphical descriptions are available as in the ATE case, though it is important to note that these comparisons all assess balance relative to the "treatment" group rather than by comparing balance for all possibly pairwise treatment group comparisons as is done with ATE.
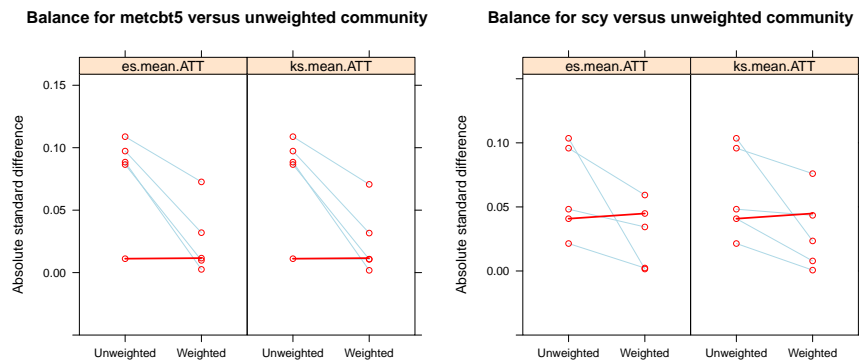
```
>       plot(mnps.AOD.ATT, plots = 1)
```
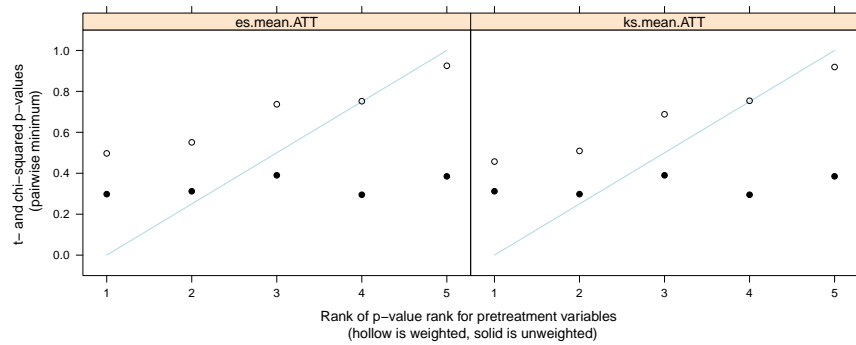

15
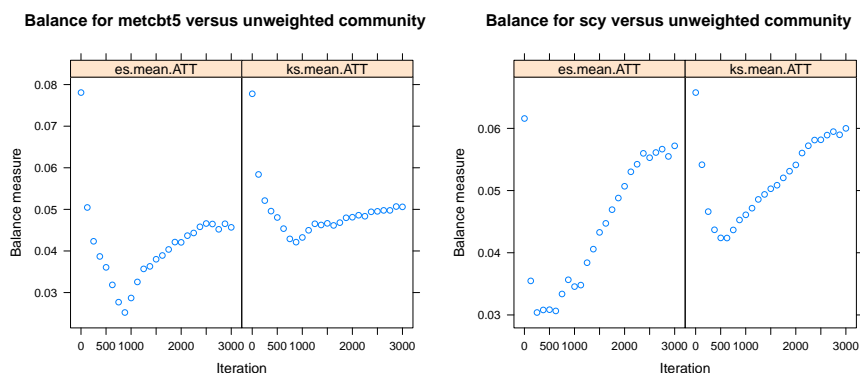
```
>       plot(mnps.AOD.ATT, plots = 3)
```



```
>       plot(mnps.AOD.ATT, plots = 3, pairwiseMax = FALSE)
```



```
>       plot(mnps.AOD.ATT, plots = 4)
```


3.3   Tabular assessments of balance

The `bal.table` output is similar to the ATE case. However, for ATT, we only report pairwise comparisons that include the `treatATT` category.

```
> bal.table(mnps.AOD.ATT, digits = 2)
```

Note that `tx' refers to the category specified as the treatATT, community.

	var	tx.mn	tx.sd	ct.mn	ct.sd	std.eff.sz	stat	p	ks	ks.pval	control
1	illact	0.10	1.04	0.01	1.03	0.09	0.87	0.38	0.10	0.27	metcbt5
2	crimjust	-0.07	1.05	0.04	1.04	-0.10	-0.98	0.33	0.10	0.22	metcbt5
3	subprob	-0.06	0.97	0.03	1.02	-0.09	-0.86	0.39	0.09	0.39	metcbt5
4	subdep	0.05	1.08	0.06	1.05	-0.01	-0.11	0.91	0.06	0.92	metcbt5
5	white	0.16	0.37	0.20	0.40	-0.11	-1.04	0.30	0.04	1.00	metcbt5
6	illact	0.10	1.04	0.12	0.96	-0.02	-0.22	0.82	0.06	0.87	scy
7	crimjust	-0.07	1.05	-0.17	1.03	0.10	1.05	0.30	0.08	0.55	scy
8	subprob	-0.06	0.97	-0.01	0.97	-0.05	-0.48	0.63	0.09	0.39	scy
9	subdep	0.05	1.08	-0.06	0.96	0.10	1.01	0.31	0.08	0.47	scy
10	white	0.16	0.37	0.18	0.38	-0.04	-0.40	0.69	0.02	1.00	scy
11	illact	0.10	1.04	0.09	1.02	0.01	0.09	0.93	0.04	1.00	metcbt5
12	crimjust	-0.07	1.05	-0.03	1.00	-0.03	-0.32	0.75	0.05	0.96	metcbt5
13	subprob	-0.06	0.97	-0.06	0.99	0.00	0.02	0.98	0.04	1.00	metcbt5
14	subdep	0.05	1.08	0.06	1.05	-0.01	-0.11	0.91	0.05	0.96	metcbt5
15	white	0.16	0.37	0.19	0.39	-0.07	-0.68	0.50	0.03	1.00	metcbt5
16	illact	0.10	1.04	0.10	1.01	0.00	-0.02	0.98	0.06	0.90	scy
17	crimjust	-0.07	1.05	-0.06	1.00	0.00	-0.02	0.99	0.05	0.94	scy
18	subprob	-0.06	0.97	-0.03	0.97	-0.03	-0.34	0.74	0.06	0.90	scy
19	subdep	0.05	1.08	-0.02	0.99	0.06	0.60	0.55	0.07	0.71	scy
20	white	0.16	0.37	0.18	0.38	-0.04	-0.43	0.67	0.02	1.00	scy
21	illact	0.10	1.04	0.09	1.02	0.01	0.10	0.92	0.04	1.00	metcbt5
22	crimjust	-0.07	1.05	-0.03	1.00	-0.03	-0.31	0.75	0.05	0.96	metcbt5
23	subprob	-0.06	0.97	-0.06	0.99	0.00	0.02	0.99	0.04	1.00	metcbt5
24	subdep	0.05	1.08	0.06	1.05	-0.01	-0.10	0.92	0.05	0.96	metcbt5
25	white	0.16	0.37	0.19	0.39	-0.07	-0.66	0.51	0.03	1.00	metcbt5
26	illact	0.10	1.04	0.10	1.04	0.00	-0.01	1.00	0.05	0.96	scy
27	crimjust	-0.07	1.05	-0.04	0.97	-0.02	-0.23	0.81	0.04	1.00	scy
28	subprob	-0.06	0.97	-0.02	0.98	-0.04	-0.40	0.69	0.04	0.99	scy
29	subdep	0.05	1.08	-0.04	0.99	0.08	0.74	0.46	0.07	0.66	scy
30	white	0.16	0.37	0.16	0.37	-0.01	-0.08	0.94	0.00	1.00	scy

	stop.method
1	unw
2	unw
3	unw
4	unw
5	unw
6	unw
7	unw
8	unw
9	unw
10	unw
11	es.mean
12	es.mean
13	es.mean
14	es.mean
15	es.mean
16	es.mean

```
17      es.mean
18      es.mean
19      es.mean
20      es.mean
21      ks.mean
22      ks.mean
23      ks.mean
24      ks.mean
25      ks.mean
26      ks.mean
27      ks.mean
28      ks.mean
29      ks.mean
30      ks.mean

> bal.table(mnps.AOD.ATT, digits = 2, collapse.to  = "covariate")

        var max.std.eff.sz min.p max.ks min.ks.pval stop.method
1    illact           0.09  0.38   0.10        0.27         unw
2  crimjust           0.10  0.30   0.10        0.22         unw
3   subprob           0.09  0.39   0.09        0.39         unw
4    subdep           0.10  0.31   0.08        0.47         unw
5     white           0.11  0.30   0.04        1.00         unw
6    illact           0.01  0.93   0.06        0.90     es.mean
7  crimjust           0.03  0.75   0.05        0.94     es.mean
8   subprob           0.03  0.74   0.06        0.90     es.mean
9    subdep           0.06  0.55   0.07        0.71     es.mean
10    white           0.07  0.50   0.03        1.00     es.mean
11   illact           0.01  0.92   0.05        0.96     ks.mean
12 crimjust           0.03  0.75   0.05        0.96     ks.mean
13  subprob           0.04  0.69   0.04        0.99     ks.mean
14   subdep           0.08  0.46   0.07        0.66     ks.mean
15    white           0.07  0.51   0.03        1.00     ks.mean

> bal.table(mnps.AOD.ATT, digits = 3, collapse.to = "stop.method")

  max.std.eff.sz min.p max.ks min.ks.pval stop.method
1          0.109 0.295  0.105       0.221         unw
2          0.073 0.497  0.069       0.707     es.mean
3          0.076 0.457  0.074       0.664     ks.mean
```

3.4   Estimating treatment effects

The process to analyze the outcome variable is also similar:

```
> require(survey)
> AOD$w.ATT <- get.weights(mnps.AOD.ATT, stop.method = "es.mean")
> design.mnps.ATT <- svydesign(ids=~1, weights=~w.ATT, data=AOD)

> glm1 <- svyglm(suf12 ~ as.factor(treat), design = design.mnps.ATT)
> summary(glm1)
```

```
Call:
svyglm(formula = suf12 ~ as.factor(treat), design = design.mnps.ATT)

Survey design:
svydesign(ids = ~1, weights = ~w.ATT, data = AOD)

Coefficients:
                        Estimate Std. Error t value
(Intercept)             -0.10505    0.06383  -1.646
as.factor(treat)metcbt5  0.20071    0.10409   1.928
as.factor(treat)scy      0.08076    0.09901   0.816
                        Pr(>|t|)
(Intercept)               0.1003
as.factor(treat)metcbt5   0.0543 .
as.factor(treat)scy       0.4150
---
Signif. codes:
0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

(Dispersion parameter for gaussian family taken to be 0.9746663)

Number of Fisher Scoring iterations: 2
```

Note in this case that the estimated treatment effect of community on those exposed to the community treatment is slightly stronger than in the ATE case (high numbers are bad for the outcome variable). Although not statistically significant, such differences are compatible with the notion that the youths who actually received the community treatment responded more favorably to it than the "average" youth would have (where the average is taken across the whole collection of youths enrolled in the study).

The discussion in McCaffrey et al. (2013) may be useful for determining whether the ATE or ATT is of greater interest in a particular application.

4 Conclusion

Often, more than two treatments are available to study participants. If the study is not randomized, analysts may be interested in using a propensity score approach. Previously, few tools existed to aide the analysis of such data, perhaps tempting analysts to ignore all but two of the treatment conditions. We hope that this extension to the twang package will encourage more appropriate analyses of observational data with more than two treatment conditions.

Acknowledgements

References

[1] Burgette, L.F., D.F. McCaffrey, B.A. Griffin (forthcoming). "Propensity score estimation with boosted regression." In W. Pan and H. Bai (Eds.) *Propensity Score Analysis: Fundamentals, Developments and Extensions.* New York: Guilford Publications, Inc.

[2] McCaffrey, D.F., B.A. Griffin, D. Almirall, M.E. Slaughter, R. Ramchand, and L.F. Burgette (2013). "A tutorial on propensity score estimation for multiple treatments using generalized boosted models." Forthcoming at *Statistics in Medicine.*

[3] Ridgeway, G., D. McCaffrey, B.A. Griffin, and L. Burgette (2014). "twang: Toolkit for weighting and analysis of non-equivalent groups." Available at http://cran.r-project.org/web/packages/twang/vignettes/twang.pdf.