cem: Coarsened exact matching in Stata

Matthew Blackwell
Harvard University
Cambridge, MA
mblackwell@iq.harvard.edu

Stefano Iacus
Universitá degli Studi di Milano
Milano, Italy
stefano.iacus@unimi.it

Gary King
Harvard University
Cambridge, MA
king@harvard.edu

Giuseppe Porro
Universitá degli Studi di Trieste
Trieste, Italy
giuseppe.porro@econ.units.it

Abstract. In this article, we introduce a Stata implementation of coarsened exact matching, a new method for improving the estimation of causal effects by reducing imbalance in covariates between treated and control groups. Coarsened exact matching is faster, is easier to use and understand, requires fewer assumptions, is more easily automated, and possesses more attractive statistical properties for many applications than do existing matching methods. In coarsened exact matching, users temporarily coarsen their data, exact match on these coarsened data, and then run their analysis on the uncoarsened, matched data. Coarsened exact matching bounds the degree of model dependence and causal effect estimation error by ex ante user choice, is monotonic imbalance bounding (so that reducing the maximum imbalance on one variable has no effect on others), does not require a separate procedure to restrict data to common support, meets the congruence principle, is approximately invariant to measurement error, balances all nonlinearities and interactions in sample (i.e., not merely in expectation), and works with multiply imputed datasets. Other matching methods inherit many of the coarsened exact matching method's properties when applied to further match data preprocessed by coarsened exact matching. The `cem` command implements the coarsened exact matching algorithm in Stata.

Keywords: st0176, cem, imbalance, matching, coarsened exact matching, causal inference, balance, multiple imputation

1 Introduction

The `cem` command is designed to improve the estimation of causal effects via a powerful method of matching that is widely applicable in observational data, and easy to understand and use (if you understand how to draw a histogram, you will understand this method). The command implements the coarsened exact matching (CEM) algorithm described in Iacus, King, and Porro (2008).

CEM is a monotonic imbalance-reducing matching method, which means that the balance between the treated and the control groups is chosen by ex ante user choice rather than being discovered through the usual laborious process of checking after the fact, tweaking the method, and repeatedly reestimating. CEM also assures that adjust-

ing the imbalance on one variable has no effect on the maximum imbalance of any other. CEM strictly bounds through ex ante user choice both the degree of model dependence and the average treatment-effect estimation error, eliminates the need for a separate procedure to restrict data to common empirical support, meets the congruence principle, is robust to measurement error, works well with multiple-imputation methods for missing data, can be completely automated, and is extremely fast computationally, even with very large datasets. After preprocessing data with CEM, the analyst may then use a simple difference in means or whatever statistical model he or she would have applied without matching.

CEM can also be used to improve other methods of matching by applying those methods to CEM-matched data (they formally inherit CEM's properties if applied within CEM strata). CEM also works well for determining blocks in randomized experiments and evaluating extreme counterfactuals.

2 Background

2.1 Notation

Consider a sample of n units randomly drawn from a population of N units, where $n \leq N$. For unit i, denote T_i as an indicator variable with the value $T_i = 1$ if unit i receives the treatment (and so is a member of the "treated" group) and the value $T_i = 0$ if it does not (and is therefore a member of the "control" group). The outcome variable is denoted Y, where $Y_i(0)$ is the potential outcome for observation i if the unit does not receive treatment and $Y_i(1)$ is the potential outcome if the (same) unit does receive treatment. For each observed unit, the observed outcome is $Y_i = T_i Y_i(1) + (1 - T_i) Y_i(0)$, and so $Y_i(0)$ is unobserved if i receives treatment and $Y_i(1)$ is unobserved if i does not receive treatment.

To compensate for the observational data problem where the treated and the control groups are not necessarily identical before treatment (and, lacking random assignment, not the same on average), matching estimators attempt to control for pretreatment covariates. For this purpose, we denote $X = (X_1, X_2, \ldots, X_k)$ as a k-dimensional dataset, where each X_j is a column vector of observed values of pretreatment variable j for the n sample observations (possibly drawn from a population of size N). That is, $X = X_{ij}$, $i = 1, \ldots, n$ and $j = 1, \ldots, k$.

2.2 Quantities of interest

As usual, the treatment effect (TE) for unit i, $\text{TE}_i = Y_i(1) - Y_i(0)$, is unobserved. All relevant causal quantities of interest are functions of TE_i, for different groups of units, and so must be estimated. We focus on the sample average treatment effect on the treated (SATT):

$$\text{SATT} = \frac{1}{n_T} \sum_{i \in T} \text{TE}_i \tag{1}$$

where $n_T = \sum_{i=1}^{n} T_i$ and $T = (1 \leq i \leq n : T_i = 1)$. Matching algorithms sometimes also change the quantity being estimated to one that can be estimated without much model dependence by selecting control or treated units.

We assume that treatment assignment is ignorable conditional on X. This assumption is often stated as "no unmeasured confounders" or "no omitted variables". Formally, this means that the treatment assignment is independent of the potential outcomes:

$$P\{T \mid X, Y(0), Y(1)\} = P(T \mid X) \tag{2}$$

2.3   Existing matching methods and practice

Matching is a nonparametric method of controlling for some of or all the confounding influence of pretreatment control variables in observational data. The key goal of matching is to prune observations from the data so that the remaining data have better *balance* between the treated and the control groups, meaning that the empirical distributions of the covariates (X) in the groups are more similar. Exactly balanced data means that controlling further for X is unnecessary (because it is unrelated to the treatment variable), and so a simple difference in means on the matched data can estimate the causal effect; approximately balanced data require controlling for X with a model (such as the same model that would have been used without matching), but the only inferences necessary are those relatively close to the data, leading to less model dependence and reduced statistical bias than without matching.

The most common matching methods involve finding, for each treated unit, at least one control unit that is "similar" on the covariates. The distinction between methods is how to define this similarity. For example, *exact matching* simply matches a treated unit to all the control units with the same covariate values. Unfortunately, because of the richness of covariates in many examples, this method often produces very few matches. A whole host of *approximate matching* methods specify a metric to find control units that are close to the treated unit. This metric is often the Mahalanobis distance or the propensity score (which is simply the probability of being treated, conditional on the covariates). Many of these related methods are implemented in Stata (Becker and Ichino 2002; Abadie et al. 2004; Leuven and Sianesi 2003; Abadie, Diamond, and Hainmueller forthcoming-a, -b). A problem with this type of solution is that it requires the user to set the size of the matching solution ex ante, and then check for balance ex post. Thus analysts must check for balance after the algorithm is finished, and then respecify a matching model and recheck balance, etc. This process repeats until the user obtains an acceptable amount of balance.

Because matching is simply a data-preprocessing technique, analysts must still apply statistical estimators to the data after matching. When one-to-one exact matching is used, a simple difference in means between Y in the treated group and the control group provides an estimator of the causal effect. When the match is not exact, a parametric model must be used to control for the differences in the covariates across the treated and the control groups. This may be a linear regression, a maximum likelihood estimator,

or some other estimator. Applying a matching method to the data before analysis can reduce the degree of model dependence (Ho et al. 2007).

One wrinkle in the analysis of matched data occurs when there are not equal numbers of treated and control units within strata. In this situation, estimators require weighting observations according to the size of their strata (Iacus, King, and Porro 2008).

3   CEM

3.1   The algorithm

The central motivation for CEM is that while exact matching provides perfect balance, it typically produces few matches because of curse-of-dimensionality issues. For instance, adding one continuous variable to a dataset effectively kills exact matching because two observations are unlikely to have identical values on a continuous measure. The idea of CEM is to temporarily coarsen each variable into substantively meaningful groups, exact match on these coarsened data, and then retain only the original (uncoarsened) values of the matched data.

Because coarsening is a process at the heart of measurement, many analysts know how to coarsen a variable into groups that preserve information. For instance, education may be measured in years, but many would be comfortable grouping observations into categories of high school, some college, college graduates, etc. This method works by exact matching on distilled information in the covariates as chosen by the user.

The algorithm works as follows:

1. Begin with the covariates X and make a copy, which we denote as X^*.

2. Coarsen X^* according to user-defined cutpoints or CEM's automatic binning algorithm.

3. Create one stratum per unique observation of X^*, and place each observation in a stratum.

4. Assign these strata to the original data, X, and drop any observation whose stratum does not contain at least one treated and one control unit.

Once completed, these strata are the foundations for calculating the treatment effect. The inherent tradeoff of matching is reflected in CEM, too: larger bins (more coarsening) used to make X^* will result in fewer strata. Fewer strata will result in more diverse observations within the same strata and, thus, higher imbalance.

It is important that CEM prunes both treated and control units. This process changes the quantity of interest under study to the treatment effect in the post matching subsample. This change is reasonable so long as the decision is transparent (see, e.g., Crump et al. [2009]).

3.2   The benefits

Iacus, King, and Porro (2008) derive many of the properties of the CEM algorithm and
we review some of them here. The key property of CEM is that it is in a class of match-
ing methods called monotonic imbalance bounding. Monotonic imbalance bounding
methods bound the maximum imbalance in some feature of the empirical distributions
through an ex ante choice by the user. In CEM, this ex ante choice is the coarsening.
As the coarsening on any variable becomes finer (the bins become more narrow), the
bound on the maximum imbalance on the moments of that variable becomes tighter.
This is also true for the bound on differences in the empirical quantiles. Furthermore,
this choice also bounds the maximum imbalance on the full multivariate histogram of
treated and control units, which includes all interactions and nonlinearities. By choos-
ing the coarsening ex ante, users can control the amount of imbalance in the matching
solution. Iacus, King, and Porro (2008) also show that CEM bounds both the error in
estimating the average treatment effect and the amount of model dependence.

Aside from bounding the imbalance between the treated and control groups, CEM
has a number of other beneficial properties. First, CEM meets the *congruence principle*,
which states that the data space and analysis space should be the same. Methods that
fail to meet this principle often produce strange or counterintuitive results. Methods
that meet the principle allow analysts to leverage their substantive knowledge of the
data to find better matches. Second, CEM automatically restricts the matched data
to areas of common empirical support. This is necessary to remove the possibility of
difficult-to-justify extrapolations of the causal effect that end up being heavily model
dependent (King and Zeng 2006). Finally, CEM is computationally very efficient, even
for large datasets.

4   An extended example

We show here how to use `cem`[1] through a simple running example: the national sup-
ported work demonstration data, also known as the LaLonde dataset (LaLonde 1986).
In this dataset, training was provided to selected individuals for 12–18 months, as was
help finding a job in the hopes of increasing the participant's earnings. The treatment
variable, `treated`, is 1 for participants (the treatment group) and 0 for nonparticipants
(the control group). The key outcome variable is earnings in 1978 (`re78`). The statistical
goal is to estimate a specific version of a causal effect: the SATT.

Because participation in the command was not assigned strictly at random, we must
control for a set of pretreatment variables by the CEM algorithm. These pretreatment

---

1. `cem` is licensed under GNU General Public License (GLP2). For more information, see
   http://gking.harvard.edu/cem/. To install `cem` in Stata 10 or later, type

   . `net from http://gking.harvard.edu/cem/`
   . `net install cem`

   In addition to the Stata version of `cem`, there is an R version in the `cem` package. The example
   presented here is also used in that package as a vignette and includes some obvious overlap in prose.

variables include age (`age`), years of education (`education`), marital status (`married`), lack of a high school diploma (`nodegree`), race (`black`, `hispanic`), indicator variables for unemployment in 1974 (`u74`) and 1975 (`u75`), and real earnings in 1974 (`re74`) and 1975 (`re75`). Some of these are dichotomous (`married`, `nodegree`, `black`, `hispanic`, `u74`, `u75`), some are categorical (`age` and `education`), and the earnings variables are continuous and highly skewed with point masses at zero. You can load these data into Stata by using the command

```
. use http://gking.harvard.edu/cem/lalonde
(Written by R.              )
```

Matching is not a method of estimation; it is a way to preprocess a dataset so that estimation of SATT based on the matched dataset will be less "model dependent" (i.e., less a function of apparently small and indefensible modeling decisions) than when based on the original full dataset. Matching involves pruning observations that have no close matches on pretreatment covariates in both the treated and the control groups. The result is typically less model dependence, lower bias, and (by removing heterogeneity) increased efficiency (King and Zeng 2006; Ho et al. 2007; Iacus, King, and Porro 2008).

4.1 Basic evaluation and analysis of unmatched data

We begin the simple difference in means as a naïve estimate of SATT; this estimator is useful only when the in-sample distribution of pretreatment covariates happens to be the same in the treatment and the control groups. First, we compute the size of the treated and the control groups:

```
. table treated
```

treated	Freq.
0	425
1	297

Thus the data include 297 treated units and 425 control units. The (unadjusted and therefore likely biased) difference in means can be found by a simple linear regression of outcome on treatment:

```
. regress re78 treated
```

Source	SS	df	MS
Model	137332528	1	137332528
Residual	2.8053e+10	720	38962865.4
Total	2.8191e+10	721	39099300.5

```
Number of obs =     722
F(  1,   720) =    3.52
Prob > F      = 0.0609
R-squared     = 0.0049
Adj R-squared = 0.0035
Root MSE      =    6242
```

re78	Coef.	Std. Err.	t	P>|t|	[95% Conf. Interval]
treated	886.3038	472.0863	1.88	0.061	-40.52625 1813.134
_cons	5090.048	302.7826	16.81	0.000	4495.606 5684.491

Thus our estimate of SATT is 886.3. Because the variable `treated` was not randomly assigned, the pretreatment covariates differ between the treated and the control groups. To see this, we focus on these pretreatment covariates: `age`, `education`, `black`, `nodegree`, and `re74`.

The overall imbalance is given by the $\mathcal{L}_1$ statistic, introduced in Iacus, King, and Porro (2008) as a comprehensive measure of global imbalance. It is based on the L_1 difference between the multidimensional histogram of all pretreatment covariates in the treated group and the same in the control group. First, we coarsen the covariates into bins. To use this measure, we require a list of bin sizes for the numerical variables. Our functions compute these automatically, or they can be set by the user.[2] Then we cross-tabulate the discretized variables as $X_1 \times \cdots \times X_k$ for the treated and the control groups separately, and record the k-dimensional relative frequencies for the treated $f_{\ell_1 \cdots \ell_k}$ and for the control $g_{\ell_1 \cdots \ell_k}$ units. Finally, our measure of imbalance is the absolute difference over all the cell values:

$$\mathcal{L}_1(f, g) = \frac{1}{2} \sum_{\ell_1 \cdots \ell_k} \mid f_{\ell_1 \cdots \ell_k} - g_{\ell_1 \cdots \ell_k} \mid \tag{3}$$

Perfect global balance (up to coarsening) is indicated by $\mathcal{L}_1 = 0$, and larger values indicate larger imbalance between the groups, with a maximum of $\mathcal{L}_1 = 1$, which indicates complete separation. If we denote the relative frequencies of a matched dataset by f^m and g^m, then a good matching solution would produce a reduction in the $\mathcal{L}_1$ statistic; that is, we would hope to have $\mathcal{L}_1(f^m, g^m) \leq \mathcal{L}_1(f, g)$.

We compute the $\mathcal{L}_1$ statistic, as well as several unidimensional measures of imbalance, via our `imbalance` command. In our running example,

```
. imbalance age education black nodegree re74, treatment(treated)
Multivariate L1 distance: .50759358

Univariate imbalance:

                  L1      mean     min      25%      50%      75%      max
       age    .10119     .1792       0        1        0       -1       -6
 education    .10047    .19224       1        0        1        1        2
     black    .00135    .00135       0        0        0        0        0
  nodegree    .08348   -.08348       0       -1        0        0        0
      re74     .0522   -101.49       0        0   69.731   584.92    -2139
```

---

2. Of course, as with drawing histograms, the choice of bins affects the final result. The crucial point is to choose one and keep it the same throughout to allow for fair comparisons. The particular choice is less crucial.

Only the overall $\mathcal{L}_1$ statistic measure includes imbalance with respect to the full joint distribution, including all interactions, of the covariates; for our example, $\mathcal{L}_1 = 0.5076$. The $\mathcal{L}_1$ value is not valuable on its own, but rather as a point of comparison between matching solutions. The value 0.5076 is a baseline reference for the unmatched data. Once we have a matching solution, we will compare its $\mathcal{L}_1$ value to 0.5076 and gauge the increase in balance due to the matching solution from that difference. Thus $\mathcal{L}_1$ works for imbalance as R^2 works for model fit: the absolute values mean less than comparisons between matching solutions. The unidimensional measures in the table are all computed for each variable separately.

The first column, labeled L1, reports the $\mathcal{L}_1^j$ measure, which is $\mathcal{L}_1$ computed for the jth variable separately (which of course does not include interactions). The second column in the table of unidimensional measures, labeled mean, reports the difference in means. The remaining columns in the table report the difference in the empirical quantiles of the distributions of the two groups for the 0th (min), 25th, 50th, 75th, and 100th (max) percentiles for each variable.

This particular table shows that variable re74 is imbalanced in the raw data in many ways and variable age is balanced in means but not in the quantiles of the two distributions. This table also illustrates the point that balancing only the means between the treated and the control groups does not necessarily guarantee balance in the rest of the distribution. Most important, of course, is the overall $\mathcal{L}_1$ measure, because even if the marginal distribution of every variable is perfectly balanced, the joint distribution can still be highly imbalanced.

4.2 CEM algorithm

We now apply the CEM algorithm by calling the cem command. The CEM algorithm performs exact matching on coarsened data to determine matches and then passes on the uncoarsened data from observations that were matched to estimate the causal effect. Exact matching works by first sorting all the observations into strata, each of which has identical values for all the coarsened pretreatment covariates, and then discarding all observations within any stratum that does not have at least one observation for each unique value of the treatment variable.

To run this algorithm, we must choose a type of coarsening for each covariate. We show how this is done via a fully automated procedure in the next section. Then we show how to use explicit prior knowledge to choose the coarsening, which is normally preferable when feasible.

In CEM, the treatment variable may be *dichotomous* or *multichotomous*. Alternatively, `cem` may be used for *randomized block experiments* without specifying a treatment variable; here the strata are simply returned without any pruning of observations.

Automated coarsening

In our running example, we have a dichotomous treatment variable. In the following code, we match on our chosen pretreatment variables, but not `re78`, which is the outcome variable and so should never be included.

The output contains useful information about the match, including a (small) table about the number of observations in total, matched, and unmatched by treatment group, as well as the results of a call to the `imbalance` command for information about the quality of the matched data. Because `cem` bounds the imbalance ex ante, the most important information is the number of observations matched. But the results also give the imbalance in the matched data by using the same measures as described in section 4.1 on the original data. Thus

```
. cem age education black nodegree re74, treatment(treated)

Matching Summary:
-----------------
Number of strata: 205
Number of matched strata: 67

            0     1
      All  425   297
  Matched  324   228
Unmatched  101    69

Multivariate L1 distance: .46113967

Univariate imbalance:
                L1       mean      min      25%      50%      75%      max
      age    .13641   -.17634        0        0        0        0       -1
education    .00687     .00687       1        0        0        0        0
    black   3.2e-16   -2.2e-16       0        0        0        0        0
 nodegree   5.8e-16    4.4e-16       0        0        0        0        0
     re74    .06787     34.438       0        0   492.23   39.425   96.881
```

We can see from these results the number of observations matched and thus retained, as well as those that were pruned because they were not comparable. By comparing the imbalance results with the original imbalance table given in the previous section, we can see that a good match can produce a substantial reduction in imbalance, not only in the means, but also in the marginal and joint distributions of the data.

The `cem` command also generates weights (stored in `cem_weights`) for use in the evaluation of imbalance measures and estimates of the causal effect.

Coarsening by explicit user choice

The power and simplicity of `cem` comes from choosing the coarsening yourself rather than using the automated algorithm as in the previous section. Choosing the coarsening enables you to set the maximum level of imbalance ex ante, which is a direct function of the coarsening you choose. By controlling the coarsening, you also put an explicit bound on the degree of model dependence and the SATT estimation error.

Fortunately, the coarsening is a fundamentally substantive act, almost synonymous with the measurement of the original variables. If you know something about the data you are analyzing, you almost surely have enough information to choose the coarsening. (And if you do not know something about the data, you might ask why you are analyzing it in the first place!)

In general, we want to set the coarsening for each variable such that substantively indistinguishable values are grouped and assigned the same numerical value. Groups may be of different sizes if appropriate. Recall that any coarsening during `cem` is used only for matching; the original values of the variables are passed on to the analysis stage for all matched observations.

For numerical variables, we can use the cutpoints syntax in `cem`. Thus, for example, in the U.S. educational system, the following discretization of years of education corresponds to different levels of school:

Grade school	0–6
Middle school	7–8
High school	9–12
College	13–16
Graduate school	>16

Using these natural breaks in the data to create the coarsening is generally a good approach and certainly better than using fixed bin sizes (as in caliper matching) that disregard these meaningful breaks. In our data, no respondents fall in the last category,

(Continued on next page)

```
. table education
```

education	Freq.
3	1
4	6
5	5
6	7
7	15
8	62
9	110
10	162
11	195
12	122
13	23
14	11
15	2
16	1

We can use the cutpoints above by using parentheses after the `education` variable:

```
. cem age education (0 6.5 8.5 12.5 16.5) black nodegree re74, treatment(treated)
Matching Summary:
-----------------
Number of strata: 155
Number of matched strata: 53

              0    1
      All   425  297
  Matched   349  245
Unmatched    76   52

Multivariate L1 distance: .43604654

Univariate imbalance:
                L1      mean     min    25%      50%      75%      max
      age    .05034  -.15556     0       0        0        1       -1
education    .0309    .00362     1      -1        0        0        2
    black   8.2e-16   1.0e-15    0       0        0        0        0
 nodegree   1.2e-15   1.9e-15    0       0        0        0        0
     re74    .04975   2.5048     0       0     161.28   -17.37   1198.1
```

As we can see, this matching solution differs from that resulting from our automated approach in the previous section. In fact, it has actually increased the balance in matching solution while giving us a higher number of matched units.

Coarsening categorical variables

For categorical variables that do not have a natural ordering, some recoding might be necessary before inputting to `cem`. For instance, if we have a variable that is

Strongly agree	1
Agree	2
Neutral	3
Disagree	4
Strongly disagree	5
No opinion	6

then there is a category ("No opinion") that does not fit on the ordinal scale of the variable. In our example dataset, we have such a variable, `q1`:

```
. table q1
```

q1	Freq.
strongly agree	121
agree	111
neutral	129
disagree	121
strongly disagree	118
no opinion	122

To coarsen this variable, first create a new coarsened variable with the `recode` command:[3]

```
. recode q1 (1 2 = 1 "agree") (3 6 = 2 "neutral") (4 5 = 3 "disagree"),
> generate(cem_q1)
(601 differences between q1 and cem_q1)
. table cem_q1
```

RECODE of q1	Freq.
agree	232
neutral	251
disagree	239

Here we have collapsed the opinions into the direction of opinion, also grouping "No opinion" with "Neutral." Once the coarsened variable is created, you can pass this variable to `cem` with the `(#0)` cutpoints argument after it to ensure that `cem` does not coarsen further:

---

3. For variables that are strictly string (nonnumeric) variables, users will need to first use the `encode` command to convert the strings to numeric and then use `recode`.

```
. cem age education black nodegree re74 cem_q1 (#0), treatment(treated)

Matching Summary:
-----------------
Number of strata: 315
Number of matched strata: 81

              0     1
      All   425   297
  Matched   260   190
Unmatched   165   107

Multivariate L1 distance: .5904067

Univariate imbalance:
                 L1      mean      min      25%      50%      75%      max
      age    .14574   -.1994        0        0        0        1       -1
education    .00263    .00263       1        0        0        0        0
    black   3.6e-16   6.7e-16       0        0        0        0        0
 nodegree   3.5e-16   6.7e-16       0        0        0        0        0
     re74    .09854    70.061       0        0    375.1  -383.76   96.881
   cem_q1   3.1e-16   3.1e-15       0        0        0        0        0
```

When calculating treatment effects after running `cem`, be sure to use the original, uncoarsened variables for analysis. Coarsened variables should only be used to produce matches. After this, they can be discarded.

4.3   Restricting the matching solution to a k-to-k match

By default, `cem` uses maximal information, resulting in strata that may include different numbers of treated and control units. To compensate for the differential strata sizes, `cem` also returns weights to be used in subsequent analyses. Although this is generally the best option, a user with enough data may opt for a k-to-k solution to avoid the slight inconvenience of needing to use weights.

The `k2k` option accomplishes this by pruning observations from a `cem` solution within each stratum until the solution contains the same number of treated and control units within all strata. Pruning occurs within a stratum (for which observations are indistinguishable to `cem` proper) by random matching inside `cem` strata.[4]

Here is an example of this approach. Running the earlier call with the `k2k` option yields the following:

---

4. In the R version of this software, pruning within strata can be done using a distance metric.

```
. cem age education black nodegree re74, treatment(treated) k2k
Matching Summary:
-----------------
Number of strata: 205
Number of matched strata: 67

               0    1
      All    425  297
  Matched    205  205
Unmatched    220   92

Multivariate L1 distance: .37560976

Univariate imbalance:
                L1      mean      min     25%      50%      75%      max
      age   .07805  -.10732       0       0        0        0       -1
education        0        0        1       0        0        0        0
   black        0        0        0       0        0        0        0
nodegree        0        0        0       0        0        0        0
    re74    .0439  -34.547        0       0   -120.7  -214.55   96.881
```

It is clear that the number of matched units has decreased after using the k2k option.

4.4 Estimating the causal effect from cem output

Using the output from cem, we can estimate the SATT by the regular Stata methods, by simply including the cem_weights. For example,

```
. regress re78 treated [iweight=cem_weights]

    Source |       SS       df       MS              Number of obs =     552
-----------+------------------------------          F(  1,   550) =    3.15
     Model | 128314324        1   128314324          Prob > F      =  0.0766
  Residual | 2.2420e+10      550  40764521.6          R-squared     =  0.0057
-----------+------------------------------          Adj R-squared =  0.0039
     Total | 2.2549e+10      551  40923414.2          Root MSE      =  6384.7

------------------------------------------------------------------------------
      re78 |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-----------+------------------------------------------------------------------
   treated |   979.1905   551.9132     1.77   0.077    -104.9252    2063.306
     _cons |    4919.49   354.7061    13.87   0.000     4222.745    5616.234
------------------------------------------------------------------------------
```

For convenience, we compute this as a regression of the outcome variable on a constant and the treatment variable, where the SATT estimate is the coefficient on the treated variable, in our case, 979.19. Any Stata command that accepts weights (aweight or iweight) can be used.

If exact matching (i.e., without coarsening) was chosen, this procedure is appropriate as is. In other situations, with some coarsening, some imbalance remains in the matched data. The remaining imbalance is strictly bound by the level of coarsening, which can be seen by any remaining variation within the coarsened bins. Thus a reasonable approach in this common situation is to attempt to adjust for the remaining imbalance via a statistical model. (Modeling assumptions for models applied to the matched data are much less consequential than they would otherwise be because cem is known to strictly

bound the level of model dependence.) To apply a statistical model to control for the remaining imbalance, we simply add variables to the regression command. For example,

```
. regress re78 treated re74 re75 [iweight=cem_weights]
```

Source	SS	df	MS		Number of obs =	552
					F(3, 548) =	5.42
Model	649651702	3	216550567		Prob > F =	0.0011
Residual	2.1899e+10	548	39961951.7		R-squared =	0.0288
					Adj R-squared =	0.0235
Total	2.2549e+10	551	40923414.2		Root MSE =	6321.5

re78	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
treated	988.083	546.5395	1.81	0.071	−85.48584	2061.652
re74	−.0174322	.1593346	−0.11	0.913	−.3304134	.2955491
re75	.3190651	.1744905	1.83	0.068	−.023687	.6618172
_cons	4287.523	393.0883	10.91	0.000	3515.378	5059.667

The user can also specify generalized linear model methods for binary, count, or other noncontinuous outcome variables by using their commands in Stata (`logit`, `poisson`, etc.) combined with the `iweight` syntax.

4.5   Matching and missing data

Almost all previous methods of matching assume the absence of any missing values. In contrast, `cem` offers two approaches to dealing with missing values (item nonresponse). In the first, where we treat missing values as one of the values of the variables, it is appropriate when "." is a valid value that is not really missing (such as when "no opinion" really means no opinion). The other is a special procedure to allow for multiply imputed data in `cem`.

Matching on missingness

If users leave missing values in the data, `cem` will coarsen the variables as normal but use "." as a separate category for each variable. Thus `cem` will match on missingness.

Matching multiply imputed data

Consider a dataset, some of which is missing, to be matched. One approach to analyzing data with missing values is *multiple imputation*, which involves creating m (usually about $m = 5$) datasets, each of which is the same as the original except that the missing values have been imputed in each. Uncertainty in the values of the missing cells is represented by variation in the imputations across the different imputed datasets (King et al. 2001).

Suppose that we have used some imputation command[5] to produce five imputed datasets, saved as `imp1.dta`–`imp5.dta`.

As an example, we added missingness to the example dataset and imputed it using Honaker, King, and Blackwell (2006).[6] If we place all the imputed datasets in the same directory and open the first, we can run `cem` with the `miname()` and `misets()` options to specify the root of the imputed datasets' filename and the number of datasets, respectively. In our example, this would be

```
. use imp1.dta, clear
(Written by R.              )
. cem age education black nodegree re74, treatment(treated) miname(imp) misets(5)
Matching Summary:
-----------------
Number of strata: 235
Number of matched strata: 76

           0    1
     All  425  297
 Matched  312  217
Unmatched 113   80

Multivariate L1 distance: .38286064

Univariate imbalance:
               L1      mean      min     25%      50%      75%      max
       age  .02132  -.07344   .19196      0        1        0       -1
 education  .01173   -.0121        1      0        0        0        0
     black  .00207    .00041       0      0        0        0        0
  nodegree  .00461   -.00092       0      0        0        0        0
      re74  .04987   -4.1404  -398.68      0    375.1   -236.7   96.881
```

---

5. There are many imputation commands from which to choose. Stata 11 includes the `mi` command to produce and analyze multiply imputed datasets under the missing at random assumption (StataCorp 2009). Although `mi` is convenient, it does not yet provide methods for imputing complex data, such as time-series data or complex survey data, often encountered in social science. We recommend other imputation procedures to handle such data. One such example is Amelia II (Honaker, King, and Blackwell 2006), which includes many features for performing imputations; see http://gking.harvard.edu/amelia. These features enable the user to include prior beliefs about individual cells, allow time series to be smooth, appropriately model panel or time-series–cross-section data, and use a robust set of imputation diagnostics and logical bounds on imputed values.

6. If users are interested in working with this example, they can access these sample files at http://gking.harvard.edu/cem/imp1.dta, etc. Once all five are downloaded, users can generate the following output. The original data file with missingness added is available at http://gking.harvard.edu/cem/lelonde.dta.

The output is identical to a normal run of `cem` and can be interpreted similarly. `cem` combines all the imputed data into one master dataset to which it assigns strata. To combine strata across imputation, `cem` chooses the strata most often assigned to an observation. This strata assignment is given to each of the imputed datasets (that is, the `cem_weights` variable is added to each of the datasets).

Now we estimate SATT via the usual multiple imputation combining formulas (averaging the point estimates and within and between variances, as usual; see King et al. [2001]), being sure to use `cem_weights`. This is simple using the `miest` command by Ken Scheve.[7] For example,

```
. miest imp reg re78 treated [aweight=cem_weights]
(Written by R.              )
(Written by R.              )
(Written by R.              )
(Written by R.              )
(Written by R.              )
Multiple Imputation Estimates

Model: regress
Dependent Variable: re78

Number of Observations: 529
----------------------------------------------------------------
         |      Coef.    Std. Err.        t         Df    P>|t|
----------------------------------------------------------------
 treated |     1269.2     557.2244      2.278      10902    0.023
   _cons |     4814.5     355.8442     13.530      22308    0.000
----------------------------------------------------------------
```

Note that we must use `aweight` instead of `iweight` (this is due to compatibility issues). One can use `miest` to implement a number of parametric models with the matching weights.

Also the `clarify` package,[8] which includes the `estsimp` command, is a useful package for analyzing multiply imputed data:

```
. estsimp reg re78 treated [iweight=cem_weights],
> mi(imp1.dta imp2.dta imp3.dta imp4.dta imp5.dta)
  (output omitted )
Regress estimates (via multiple imputation)          Nobs = 528

----------------------------------------------------------------
    re78 |      Coef.    Std. Err.        t        d.f.    P>|t|
---------+------------------------------------------------------
 treated |   1269.205    557.7437      2.276      10943    0.023
   _cons |    4814.52    356.1777     13.517      22392    0.000
----------------------------------------------------------------

Number of simulations  : 1000
Names of new variables : b1 b2 b3
Datasets used for MI   : imp1.dta imp2.dta imp3.dta imp4.dta imp5.dta
```

Here we were able to use `iweight` as in our earlier analyses.[9]

---

7. `miest` is available at http://gking.harvard.edu/amelia/amelia1/docs/mi.zip.

8. The `clarify` package is available at http://gking.harvard.edu/stats.shtml#clarify.

9. In addition to these commands, Stata 11 users can use the `mi estimate` command with the appropriate weights.

4.6 Blocking in randomized experiments

The cem command can produce strata for a block randomized design for a set of pre-treatment covariates. Because block randomized designs outperform complete randomization on bias, efficiency, power, and robustness, they should be used whenever possible (Imai, King, and Nall 2009; Imai, King, and Stuart 2008). To create a set of strata for a block randomized design, simply run cem without passing a treatment variable. This will assign observations to strata based on their coarsened values and create a cem_strata variable indicating this assignment. Once this is complete, simply randomly assign treatment within these strata to complete the block randomized design.

4.7 Using cem to improve other matching methods

Even if you plan to use a different matching method, you can still use the cem command to improve that matching solution. An important step before matching is restricting the data to areas of common empirical support. This helps you avoid making inference based on extrapolation, because such inferences are known to be extremely model dependent. Traditional matching methods, however, are not equipped to handle this situation. For example, the propensity score can be used to find the area of extrapolation only after we know that the correct propensity score model has been used. However, the only way to verify that the correct propensity score model has been specified is to check whether matching on it produces balance between the treated and the control groups on the relevant covariates. But balance cannot be reliably checked until the region of extrapolation has been removed. To avoid this type of infinite regress, researchers use entirely different technologies for the first step, such as kernel density estimation (Heckman, Ichimura, and Todd 1997) or dropping control units outside the hyper-rectangle (Porro and Iacus 2009) or convex hull (King and Zeng 2006) of the treated units.

The matching methods currently in Stata all rely on propensity score methods for restricting the data to common empirical support. For cem, on the other hand, this restriction is a natural consequence of the algorithm. All observations within a stratum containing both a treated and a control unit are by definition inside of the common support. In light of this, a good use of cem would be to reduce the data to common support before applying another matching solution such as psmatch2, nnmatch, or pscore. This will improve the quality of the inferences drawn from these methods. Once you have run cem, all you must do is run the following command to restrict the data to common support:

```
. drop if cem_matched == 0
```

Alternatively, you may use any of the matching methods with an if cem_matched==1 option. This will force the other matching methods to only match in the region of common support. As an example using nnmatch, this would be

```
. nnmatch re78 treated age education black nodegree re74 if cem_matched == 1
```

Of course, you may apply this idea to any matching method in Stata, not just the ones listed here.

5   The cem command

5.1   Syntax

cem *varname1* $\big[$ (*cutpoints1*) $\big]$ $\big[$ *varname2* $\big[$ (*cutpoints2*) $\big]$ $\big]$ ...

    $\big[$, <u>t</u>reatment(*varname*) <u>show</u>breaks <u>auto</u>cuts(*string*) k2k imbbreaks(*string*)

   miname(*string*) misets(*#*) $\big]$

5.2   Description

cem implements the CEM method described in Iacus, King, and Porro (2008). The main inputs for cem are the variables to use (*varname#*) and the cutpoints that define the coarsening (*cutpoints#*). The latter argument is set in parentheses after the name of the relevant variable. Users can either specify cutpoints for a variable or allow cem to automatically coarsen the data based on an automatic coarsening algorithm, chosen by the user. To specify a set of cutpoints for a variable, place a numlist in parentheses after the variable's name. To specify an automatic coarsening, place a string indicating the binning algorithm to use in parentheses after the variable's name. To create a certain number of equally spaced cutpoints including the extreme values, say, 10, place #10 in the parentheses (using #0 will force cem into not coarsening the variable at all). Omitting the parenthetical statement after the variable name tells cem to use the default binning algorithm, itself set by autocuts().

Character variables are ignored by cem. These variables will need to be converted into numeric variables by using encode. Coarsening variables that are not ordinal must be done before running cem by using the recode command, as described above.

5.3   Options

treatment(*varname*) sets the treatment variable used for matching. If omitted, cem will simply sort the observations into strata based on the coarsening and not return any output related to matching.

showbreaks displays the cutpoints used for each variable.

autocuts(*string*) sets the default automatic coarsening algorithm. The default is autocuts(sturges). Any variable without a *cutpoints#* argument after its name will use the autocuts() option.

k2k, by randomly dropping observations, produces a matching result that has the same number of treated and control units in each matched strata.

imbbreaks(*string*) sets the coarsening method for the imbalance checks printed after cem runs. This should match whichever method is used for imbalance checks elsewhere. If either cem or imbalance has been run and there is a r(L1_breaks) available, this will be the default. Otherwise, the default is imbbreaks(scott).

miname(*string*) specifies the root of the filenames of the imputed dataset. They should be in the working directory. For example, if you specified miname(imputed), then the filenames would be imputed1.dta, imputed2.dta, and so on.

misets(#) specifies the number of imputed datasets to be used for matching.

5.4 Output

The output cem returns depends on the inclusion of a treatment variable. If the treatment variable is provided, cem will match and return the following three variables in the current dataset:

cem_strata is the strata number assigned to each observation by cem.

cem_matched is 1 for a matched observation and 0 for an unmatched observation.

cem_weights is the weight of the stratum for each observation. Strata with unmatched units are given a weight of 0, and treated observations are given a weight of 1.

When using the multiple-imputation features, cem outputs the cem_treat variable, which is the treatment vector used for matching. cem applies the same combination rule to treatment as to strata.

If the options for multiple imputation are used, cem saves each of these variables in each of the imputed datasets, allowing for easy use in commands like miest.

The following are stored as saved results in Stata's memory:

Scalars
    r(n_strata)   number of strata
    r(n_groups)   number of treatment levels
    r(n_mstrata)   number of strata with matches
    r(n_matched)   number of matched observations
    r(L1)   multivariate imbalance measure

Macros
    r(varlist)   covariate variables used
    r(treatment)   treatment variable
    r(cem_call)   call to cem
    r(L1_breaks)   break method used for L1 distance

Matrices
    r(match_table) table of treatment versus matched
    r(groups)   tabulation of treatment variable
    r(imbal)   univariate imbalance measures

If the treatment variable is omitted (e.g., for blocking), then the only outputs are cem_strata, r(n_strata), r(varlist), and r(cem_call).

6   The imbalance command

6.1   Syntax

imbalance *varlist* $\big[\,if\,\big]$ $\big[\,in\,\big]$ $\big[$, <u>tr</u>eatment(*varname*) breaks(*string*)

   miname(*string*) misets(*#*) <u>use</u>weights $\big]$

6.2   Description

imbalance returns a number of measures of imbalance in covariates between treatment
and control groups. A multivariate L1 distance, univariate L1 distances, and difference
in means and empirical quantiles difference are reported. The L1 measures are computed
by coarsening the data according to breaks() and comparing across the multivariate
histogram. See Iacus, King, and Porro (2008) for more details on this measure.

6.3   Options

treatment(*varname*) sets the treatment variable used for the imbalance checks.

breaks(*string*) sets the default automatic coarsening algorithm. If cem or imbalance
   has been run and there is a r(L1_breaks) available, this will be the default. Other-
   wise, the default is breaks(scott). It is not incredibly important which method is
   used here as long as it is consistent.

miname(*string*) specifies the root of the filenames of the imputed dataset. They should
   be in the working directory. For example, if you specified miname(imputed), then
   the filenames would be imputed1.dta, imputed2.dta, and so on.

misets(*#*) sets the number of imputed datasets to be used for matching.

useweights makes imbalance use the weights from the output of cem. This is useful
   for checking balance after running cem.

6.4   Saved results

Scalars
   r(L1)   multivariate imbalance measure
Macros
   r(L1_breaks)   break method used for L1 distance
Matrices
   r(imbal)   univariate imbalance measures

7 References

Abadie, A., A. Diamond, and J. Hainmueller. Forthcoming-a. Synthetic control methods for comparative case studies: Estimating the effect of California's tobacco control program. *Journal of the American Statistical Association*.

———. Forthcoming-b. Synth: An R package for synthetic control methods in comparative case studies studies: Estimating the effect of California's tobacco control program. *Journal of Statistical Software*.

Abadie, A., D. Drukker, J. Leber Herr, and G. W. Imbens. 2004. Implementing matching estimators for average treatment effects in Stata. *Stata Journal* 4: 290–311.

Becker, S. O., and A. Ichino. 2002. Estimation of average treatment effects based on propensity scores. *Stata Journal* 2: 358–377.

Crump, R. K., V. J. Hotz, G. W. Imbens, and O. A. Mitnik. 2009. Dealing with limited overlap in estimation of average treatment effects. *Biometrika* 96: 187–199.

Heckman, J. J., H. Ichimura, and P. Todd. 1997. Matching as an econometric evaluation estimator: Evidence from evaluating a job training program. *Review of Economic Studies* 64: 605–654.

Ho, D. E., K. Imai, G. King, and E. A. Stuart. 2007. Matching as nonparametric preprocessing for reducing model dependence in parametric causal inference. *Political Analysis* 15: 199–236.

Honaker, J., G. King, and M. Blackwell. 2006. Amelia II: A program for missing data. http://gking.harvard.edu/amelia/docs/amelia.pdf.

Iacus, S. M., G. King, and G. Porro. 2008. Matching for causal inference without balance checking. http://gking.harvard.edu/files/cem.pdf.

Imai, K., G. King, and C. Nall. 2009. The essential role of pair matching in cluster-randomized experiments, with application to the Mexican Universal Health Insurance Evaluation. *Statistical Science* 24: 29–53.

Imai, K., G. King, and E. A. Stuart. 2008. Misunderstandings between experimentalists and observationalists about causal inference. *Journal of the Royal Statistical Society, Series A* 171: 481–502.

King, G., J. Honaker, A. Joseph, and K. Scheve. 2001. Analyzing incomplete political science data: An alternative algorithm for multiple imputation. *American Political Science Review* 95: 49–69.

King, G., and L. Zeng. 2006. The dangers of extreme counterfactuals. *Political Analysis* 14: 131–159.

LaLonde, R. J. 1986. Evaluating the econometric evaluations of training programs. *American Economic Review* 76: 604–620.

Leuven, E., and B. Sianesi. 2003. PSMATCH2: Stata module to perform full Mahalanobis and propensity score matching, common support graphing, and covariate imbalance testing. Statistical Software Components S432001, Department of Economics, Boston College. http://ideas.repec.org/c/boc/bocode/s432001.html.

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

StataCorp. 2009. *Stata 11 Multiple-Imputation Reference Manual.* College Station, TX: Stata Press.

About the authors

Matthew Blackwell is a doctoral candidate in the Department of Government at Harvard University and is an affiliate of the Institute for Quantitative Social Science.

Stefano M. Iacus is an associate professor in the Department of Economics, Business, and Statistics at the Universitá degli Studi di Milano, Italy.

Gary King is the Albert J. Weatherhead III University Professor and Director of the Institute for Quantitative Social Science at Harvard University.

Giuseppe Porro is an associate professor of Economic Policy in the Department of Economics and Statistics at the Universitá degli Studi di Trieste, Italy.