ebalance: A **Stata** Package for Entropy Balancing

Jens Hainmueller
Massachusetts Institute of Technology

Yiqing Xu
Massachusetts Institute of Technology

Abstract

The Stata package **ebalance** implements entropy balancing, a multivariate reweighting method described in Hainmueller (2012) that allows users to reweight a dataset such that the covariate distributions in the reweighted data satisfy a set of specified moment conditions. This can be useful to create balanced samples in observational studies with a binary treatment where the control group data can be reweighted to match the covariate moments in the treatment group. Entropy balancing can also be used to reweight a survey sample to known characteristics from a target population.

Keywords: causal inference, reweighting, matching, Stata.

1. Introduction

Methods such as nearest neighbor matching or propensity score techniques have become popular in the social sciences in recent years to preprocess data prior to the estimation of causal effects in observational studies with binary treatments under the selection on observables assumption (Ho, Imai, King, and Stuart 2007; Sekhon 2009). The goal in preprocessing is to adjust the covariate distribution of the control group data by reweighting or discarding of units such that it becomes more similar to the covariate distribution in the treatment group. This preprocessing step can reduce model dependency for the subsequent analysis of treatment effects in the preprocessed data using standard methods such as regression analysis (Abadie and Imbens 2011).

One important issue with many commonly used matching or propensity score adjustments is that they are somewhat tedious to use and often result in rather low levels of covariate balance in practice. Researchers often go back and forth between propensity score estimation, matching, balance checking to "manually" search for a suitable weighting that balances the covariate distributions. This indirect search process often fails to jointly balance out all of the covariates and in some cases even counteracts bias reduction when balance on some

covariates decreases as a result of the preprocessing (Diamond and Sekhon 2006; Iacus, King, and Porro 2012). Entropy balancing, a method described in Hainmueller (2012), addresses these shortcomings and uses a preprocessing scheme where covariate balance is directly built into the weight function that is used to adjust the control units.

Borrowing from similar methods in the literature on survey adjustments (Deming and Stephan 1940; Ireland and Kullback 1968; Zaslavsky 1988; Särndal and Lundström 2006), entropy balancing is based on a maximum entropy reweighting scheme that enables users to fit weights that satisfy a potentially large set of balance constraints that involve exact balance on the first, second, and possibly higher moments of the covariate distributions in the treatment and the reweighted control group. Instead of checking for covariate balance after the preprocessing, the user starts by specifying a desired level of covariate balance using a set of balance conditions. Entropy balancing then finds a set of weights that satisfies the balance conditions and remains as close as possible (in an entropy sense) to uniform base weights to prevent loss of information and retain efficiency for the subsequent analysis.

For users, the entropy balancing scheme has several advantages. Since the weights are directly adjusted to the known sample moments, the scheme always (at least weakly) improves on the covariate balance achieved by conventional preprocessing methods for the specified moment constraints. Balance checking is therefore no longer necessary for the included moments. Since the entropy balancing weights vary smoothly across units, they also commonly retain more information in the preprocessed data than other approaches such as nearest neighbor matching which either match or discard each control unit. The reweighting scheme is also computationally attractive; for moderate sized datasets the weights are often attained within a few seconds (if the balance constraints are feasible). Finally, entropy balancing is fairly flexible. The procedure can also be combined with other matching methods and the resulting weights are compatible with many standard estimators for subsequent analysis of the reweighted data. Apart from observational studies with binary treatments, entropy balancing methods can also be used to adjust survey samples to known characteristics of some target population.

This paper introduces a Stata (StataCorp. 2011) package called **ebalance** which implements the entropy balancing method as described in Hainmueller (2012). This package is distributed through the Statistical Software Components (SSC) archive – often called the Boston College Archive – at `http://ideas.RePEc.org/c/boc/bocode/s457326.html`.[1] The key function in the **ebalance** package is `ebalance` which allows users to fit the entropy balancing weights and offers various options to specify the balance constraints. We illustrate the use of this function with the well known LaLonde data (LaLonde 1986) from the National Supported Work Demonstration program. This data is contained in the file `cps1re74.dta` and "ships" with the **ebalance** package.

2. Entropy balancing

2.1. Motivation

Entropy balancing is based on a maximum entropy reweighting scheme that allows user to preprocess data in observational studies with binary treatments. Hainmueller (2012) provides

---

[1]We thank the editor Christopher F. Baum for managing the SSC archive. A similar software implementation of entropy balancing for R (R Core Team 2013) is available as the **ebal** package (Hainmueller 2013).

a detailed discussion of the theoretical properties and numerical implementation of the method and presents various simulations and real data examples. Here we focus on how users can implement entropy balancing using the **ebalance** package and therefore only provide a brief review of the material in Hainmueller (2012).

Imagine we have an observational study with a sample of n_1 treated and n_0 control units that are randomly drawn from populations of size N_1 and N_0 respectively ($n_1 \leq N_1$ and $n_0 \leq N_0$). Let $D_i \in \{1, 0\}$ be a binary treatment indicator coded 1 or 0 if unit i is exposed to the treatment or control condition respectively. Let X be a matrix that contains the data of J exogenous pre-treatment covariates; X_{ij} denotes for unit i the value of the j-th covariate characteristic such that $X_i = [X_{i1}, X_{i2}, ..., X_{iJ}]$ refers to the row vector of characteristics for unit i and X_j refers to the column vector with the j-th covariate. The densities of the covariates in the treatment and control population are given by $f_{X|D=1}$ and $f_{X|D=0}$ respectively. Following the potential outcome framework for causal inference, $Y_i(D_i)$ denotes the pair of potential outcomes for unit i given the treatment and control condition and observed outcomes are given by $Y = Y(1) D + (1 - D)Y(0)$.

As is common in the literature on preprocessing methods, we focus on the population average treatment effect on the treated (PATT) given by $\tau = \mathbf{E}[Y(1)|D = 1] - \mathbf{E}[Y(0)|D = 1]$. The first expectation can be directly identified from the treatment group data, but the second expectation is counterfactual, i.e., the expected outcome for the treated units in the absence of the treatment. Rosenbaum and Rubin (1983) show that assuming selection on observables, $Y(0) \perp\!\!\!\perp D|X$, and overlap, $\mathbf{Pr}(D = 1|X = x) < 1$ for all x in the support of $f_{X|D=1}$, the PATT is identified as:

$$\tau = \mathbf{E}[Y|D = 1] - \int \mathbf{E}[Y|X = x, D = 0] f_{X|D=1}(x) dx \tag{1}$$

In order to estimate the last term in Equation 1, the covariate adjusted mean, the covariate distribution in the control group data needs to be adjusted to make it similar to the covariate distribution in the treatment group data such that the treatment indicator D becomes closer to being orthogonal to the covariates. A variety of data preprocessing methods such as nearest neighbor matching, coarsened exact matching, propensity score matching, or propensity score weighting have been proposed to reduce the imbalance in the covariate distributions. Once the covariate distributions are adjusted, standard analysis methods such as regression can be subsequently used to estimate treatment effects with lower error and model dependency (Imbens 2004; Rubin 2006; Ho *et al.* 2007; Iacus *et al.* 2012; Sekhon 2009).

2.2. Entropy balancing scheme

Consider the simplest case where the treatment effect in the preprocessed data is estimated using the difference in mean outcomes between the treatment and adjusted control group. One popular preprocessing methods is to use propensity score weighting (Hirano and Imbens 2001; Hirano, Imbens, and Ridder 2003) where the counterfactual mean is estimated as

$$\mathbf{E}[Y(0)\widehat{|D} = 1] = \frac{\sum_{\{i|D=0\}} Y_i d_i}{\sum_{\{i|D=0\}} d_i} \tag{2}$$

and every control unit receives a weight given by $d_i = \frac{\hat{p}(x_i)}{1 - \hat{p}(x_i)}$. $\hat{p}(x_i)$ in Equation 2 is a propensity score that is commonly estimated with a logistic or probit regression of the treatment indicator on the covariates. If the propensity score model is correctly specified, then

the estimated weights d_i will ensure that the covariate distribution of the reweighted control units will match the covariate distribution in the treatment group. However, in practice this approach often fails to jointly balance all the covariates because the propensity score model may be misspecified. To tackle this problem researchers often go back and forth between logistic/probit regression estimation, weighting, and balance checking to search for a weighting that balances the covariates. This indirect search process is rather time-consuming and often researchers are left with low levels of covariate balance.

Entropy balancing generalizes the propensity score weighting approach by estimating the weights directly from a potentially large set of balance constraints which exploit the researcher's knowledge about the sample moments. In particular, the counterfactual mean may be estimated by

$$\mathbf{E}[Y\widehat{(0)|}D = 1] = \frac{\sum_{\{i|D=0\}} Y_i\, w_i}{\sum_{\{i|D=0\}} w_i} \tag{3}$$

where w_i is the entropy balancing weight chosen for each control unit. These weights are chosen by the following reweighting scheme that minimizes the entropy distance metric

$$\min_{w_i} H(w) = \sum_{\{i|D=0\}} w_i\, log(w_i/q_i) \tag{4}$$

subject to balance and normalizing constraints

$$\sum_{\{i|D=0\}} w_i\, c_{ri}(X_i) = m_r \quad \text{with} \quad r \in 1, ..., R \quad \text{and} \tag{5}$$

$$\sum_{\{i|D=0\}} w_i = 1 \quad \text{and} \tag{6}$$

$$w_i \geq 0 \quad \text{for all} \quad i \quad \text{such that} \quad D = 0 \tag{7}$$

where $q_i = 1/n_0$ is a base weight and $c_{ri}(X_i) = m_r$ describes a set of R balance constraints imposed on the covariate moments of the reweighted control group.

The `ebalance` function implements this reweighting scheme. The user starts by choosing the covariates that should be included in the reweighting. For each covariate, the user then specifies a set of balance constraints (in Equation 5) to equate the moments of the covariate distribution between the treatment and the reweighted control group. The moment constraints may include the mean (first moment), the variance (second moment), and the skewness (third moment). A typical balance constraint is formulated such that m_r contains the r-th order moment of a specific covariate X_j for the treatment group and the moment function is specified for the control group as $c_{ri}(X_{ij}) = X_{ij}^r$ or $c_{ri}(X_{ij}) = (X_{ij} - \mu_j)^r$ with mean μ_j. In the `ebalance` function, the balance constraints can be flexibly specified with the `targets(numlist)` option (see examples below). The user can chose to adjust the first, second, or third moments of each covariate. As we show below, comoments of the covariates can also be included in the balance constraints by including interaction terms such that for example the mean of one covariate is balanced across subgroups of another covariate.

The entropy balancing scheme then searches for a set of unit weights $W = [w_i, ..., w_{n_0}]^\top$ which minimizes Equation 4, the entropy distance between W and the vector of base weights $Q = [q_i, ..., q_{n_0}]^\top$, subject to the balance constraints in Equation 5, the normalization constraint in Equation 6, and the non-negativity constraint in Equation 7. This ensures that the weights are adjusted as far as is needed to accommodate the balance constraints, but at the same

time the weights are kept as close as possible to the uniformly distributed base weights to retain information in the reweighted data (the loss function is non-negative and decreases the closer W is to Q; the loss equals zero iff $W = Q$).[2]

The entropy balancing scheme has the advantage that it directly incorporates the auxiliary information about the known sample moments and adjusts the weights such that the user obtains exact covariate balance for all moments included in the reweighting scheme. This obviates the need for time-consuming search over logistic or probit propensity score models to find a suitable balancing solution. By including a potentially large set of balance conditions, the user can adjust the covariate density of the reweighted control group such that it becomes very similar to that in the treatment group and also rule out the possibility that balance decreases on any of the specified moments.

After the entropy balancing weights are fitted, they can be passed to any standard estimator for the subsequent analysis in the reweighted data. This can be easily accomplished in Stata using for example the suite of svy estimation commands for the analysis of weighted data.

2.3. Numerical implementation

At a first glance, numerically solving the entropy balancing reweighting scheme seems daunting given its high dimensionality (i.e., we need to find one weight for each control unit). However, as described in Hainmueller (2012) we can exploit several structural features that greatly facilitate the minimization problem. The loss function is globally convex such that a unique solution exists if the constraints are consistent. Moreover, by applying a Lagrangian and exploiting duality (Erlander 1977) the weights that solve the entropy balancing scheme can be computed from a dual problem that is unconstrained and reduced to a system of non-linear equations in R Lagrange multipliers. In particular, let $Z = \{\lambda_1, ..., \lambda_R\}^\top$ be a vector of Lagrange multipliers for the balance constraints and rewrite the constraints in matrix form as $CW = M$ with the $(R \times n_0)$ constraint matrix $C = [c_1(X_i), ..., c_R(X_i)]^\top$ and moment vector $M = [m_1, ..., m_R]^\top$.[3] The dual problem is then given by

$$\min_Z L^d = \log(Q^\top \exp(-C^\top Z)) + M^\top Z \tag{8}$$

and the vector Z^* that solves the dual problem also solves the primal problem. The solution weights can be recovered using

$$W^* = \frac{Q \cdot \exp(-C^\top Z^*)}{Q^\top \exp(-C^\top Z^*)}. \tag{9}$$

To solve the dual problem we use a Levenberg-Marquardt scheme that makes use of second order information by iterating

$$Z^{new} = Z^{old} - l \, \nabla_Z^2 (L^d)^{-1} \nabla_Z (L^d) \tag{10}$$

where l denotes the step length. In each iteration take the full Newton step or otherwise backtrack in the Newton direction to find the optimal l through a line search.

---

[2]As described in Hainmueller (2012), apart from the entropy metric we could use other distance metrics from the Cressie-Read family instead. However, we prefer the entropy metric because it generates non-negative weights, facilitates the optimization, and is also more robust to misspecification.

[3]$C^\top$ must be full column rank otherwise there exists no feasible solution.

3. Implementing entropy balancing

In this section we describe how users can implement the entropy balancing method using the **ebalance** package.

3.1. Installation

ebalance can be installed from the Statistical Software Components (SSC) archive by typing

```
. ssc install ebalance, all replace
```

on the `Stata` command line. A dataset associated with the package, `cps1re74.dta`, will be downloaded to the default `Stata` folder when option `all` is specified.

3.2. Data

We illustrate the use of **ebalance** with data from the National Supported Work Demonstration (NSW), a randomized evaluation of a subsidized work program that was first analyzed by LaLonde (1986) and has subsequently been widely used in the causal inference literature to evaluate different methods. The data contained in `cps1re74.dta` is a subset of the original LaLonde data first used by Dehejia and Wahba (1999). The data contains 185 program participants from a randomized evaluation of the NSW program, and 15,992 non-experimental non-participants drawn from the Current Population Survey Social Security Administration File (CPS-1). We refer to these groups as "treated" and "control" units respectively (notice that only "treated" units are included from the experimental data). The dataset includes 12 variables for each observation:

- `treat`: indicator for treatment status (1 if treated with NSW, 0 if control);

- `age`: age in years;

- `educ`: years of schooling;

- `black`: indicator for black;

- `hisp`: indicator for hispanic;

- `married`: indicator for married;

- `nodegree`: indicator for no high school diploma;

- `re74`: real earnings in 1974 (US Dollars);

- `re75`: real earnings in 1975 (US Dollars);

- `u74`: indicator for unemployment in 1974 (i.e., `re74` is zero);

- `u75`: indicator for unemployment in 1974 (i.e., `re74` is zero);

- `re78`: real earnings in 1978 (US Dollars).

The outcome of interest is `re78`, which measures earnings in the period after the NSW intervention. All other covariates are measured prior to the intervention. By comparing the difference in means of `re78` in the NSW experimental data, one finds that the program on average raised earnings by USD $1,794$ with a 95% confidence interval of USD $[551; 3,038]$ (see Dehejia and Wahba 1999 for details). This unbiased estimate of the average treatment effect from the experimental data is our target answer.

When using a regression of `re78` on `treat` and all covariates in the `cps1re74.dta` data with the non-experimental control group, we find that the average treatment effect is estimated at USD 1,016.

```
. use cps1re74.dta, clear
. reg re78 treat age-u75

    Source |       SS       df       MS              Number of obs =   16177
-----------+------------------------------           F( 11, 16165) = 1343.88
     Model |  7.2418e+11     11  6.5835e+10          Prob > F      =  0.0000
  Residual |  7.9190e+11  16165  48988567.3          R-squared     =  0.4777
-----------+------------------------------           Adj R-squared =  0.4773
     Total |  1.5161e+12  16176  93724175.2          Root MSE      =  6999.2


------------------------------------------------------------------------------
      re78 |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-----------+------------------------------------------------------------------
     treat |   1067.546   554.0595     1.93   0.054    -18.47193    2153.564
       age |  -94.54102   6.000283   -15.76   0.000    -106.3022    -82.7798
      educ |   175.2255   28.69658     6.11   0.000      118.977     231.474
     black |  -811.0888   212.8488    -3.81   0.000    -1228.296   -393.8815
    hispan |  -230.5349   218.6098    -1.05   0.292    -659.0344    197.9646
   married |   153.2284   142.7748     1.07   0.283     -126.626    433.0828
  nodegree |   342.9265   177.8778     1.93   0.054    -5.733561    691.5866
      re74 |   .2914332   .0127311    22.89   0.000     .2664789    .3163875
      re75 |   .4426945   .0128868    34.35   0.000      .417435     .467954
       u74 |   355.5564   231.6004     1.54   0.125    -98.40599    809.5189
       u75 |  -1612.758    239.803    -6.73   0.000    -2082.798   -1142.717
     _cons |    5762.18   445.6145    12.93   0.000     4888.726    6635.634
------------------------------------------------------------------------------
```

This indicates that the OLS estimate, which includes covariates that researchers would typically control for when evaluating the program impact, is substantially lower than the true average treatment effect established from the experimental data. Below we consider if preprocessing the data using entropy balancing allows us to more accurately recover the experimental target answer.

3.3. Basic syntax

The basic syntax of the `ebalance` function follows the standard Stata command form

```
ebalance [treat] covar [if] [in] [, options]
```

By default, `ebalance` assumes that the user has data for both a treatment and a control group. Given this two-group setup, `ebalance` will reweight the data from the control units to match a set of moments that is computed from the data of the treated units (further below we discuss how `ebalance` can be used with a single group). `treat` specifies the binary treatment indicator variable, whose values should be coded as 1 for treated and 0 for control units. `covar` specifies the list of covariates that are to be included in the entropy balancing adjustment.

The most important option in `ebalance` is `targets(numlist)`. It allows users to specify the balance constraints for the covariates included in the `covar` variable list. The user specifies a number (1, 2, or 3) which corresponds to the highest covariate moment that should be adjusted for each covariate. For example, by coding

```
. ebalance treat age black educ, targets(1)
```

the user requests that the first moments of the variables `age`, `black`, and `educ` are adjusted. Accordingly, `ebalance` computes the means of these covariates in the treatment group data (`treat==1`) and searches for a set of entropy weights such that the means in the reweighted control group data match the means from the treatment group (if the `targets` option is not specified then `targets(1)` is assumed by default). The command returns

```
Data Setup
Treatment variable:   treat
Covariate adjustment: age black educ

Optimizing...
Iteration 1: Max Difference = 53131.2184
Iteration 2: Max Difference = 19545.2011
.
Iteration 15: Max Difference = .000363868
maximum difference smaller than the tolerance level; convergence achieved

Treated units: 185     total of weights: 185
Control units: 15992   total of weights: 185

Before: without weighting
```

		Treat			Control	
	mean	variance	skewness	mean	variance	skewness
age	25.82	51.19	1.115	33.23	122	.3478
black	.8432	.1329	-1.888	.07354	.06813	3.268
educ	10.35	4.043	-.7212	12.03	8.242	-.4233

```
After:  _webal as the weighting variable
```

		Treat			Control	
	mean	variance	skewness	mean	variance	skewness

```
--------+-------------------------------+-------------------------------
    age |    25.82     51.19     1.115 |    25.82     82.54     1.158
  black |    .8432     .1329    -1.888 |    .8431     .1323    -1.887
   educ |    10.35     4.043    -.7212 |    10.35      8.03    -.8269
```

which indicates that after the entropy balancing step, the means in the reweighted control group match the means in the treatment group.

By default, `ebalance` stores the solution weights in a variable named `_webal`. If the user wants to store the fitted weights in a different variable, he can simply specify the desired variable name in the `generate(varname)` option. The entropy balancing weights can be readily used for subsequent analysis using for example the `aweight` or `svy` commands provided in `Stata` to analyze weighted data. For example, to verify that the means of `age` match in the reweighted data we can code

```
. tabstat age [aweight=_webal], by(treat) s(N me v) nototal
```

```
Summary for variables: age
     by categories of: treat (1 if treated, 0 control)

   treat |         N       mean   variance
---------+-------------------------------
       0 |     15992   25.81647   82.53714
       1 |       185   25.81622    51.1943
-----------------------------------------
```

The `targets(numlist)` option can also be used to flexibly specify higher-order balance constraints. If only a single number is specified, then that moment order will be applied to all covariates. For example, coding

```
. ebalance treat age black educ, targets(3)
```

specifies that the 1st, 2nd, and 3rd moments for all three covariates will be adjusted. Alternatively, the user can specify constraints specific to each covariate. For example, coding

```
. ebalance treat age black educ, targets(3 1 2)
```

specifies that `ebalance` will adjust the 1st, 2nd, and 3rd moment for `age`, the 1st moment for `black`, and the 1st and 2nd moment for `black`. We obtain

```
.
After: _webal as the weighting variable
```

| | | Treat | | | Control | |
	mean	variance	skewness	mean	variance	skewness
age	25.82	51.19	1.115	25.82	51.2	1.115
black	.8432	.1329	-1.888	.8432	.1322	-1.888
educ	10.35	4.043	-.7212	10.35	4.043	-.7193

which shows that after the adjustment the mean, variance, and skewness of `age` is the same in the treatment and reweighted control group. Notice that for binary variables, such as `black`, adjusting only the first moment is sufficient to match the higher moments. We also see that for `educ`, simply adjusting the means and variances results in a skewness that is almost identical between the two groups.

3.4. Interactions

`ebalance` also allows users to adjust comoments of the joint distribution of the covariates. For example, to adjust the control group data such that the mean of `age` is similar for blacks and non-blacks we can simply include an interaction term between `age` and `black` in the `covar` variable list. We code

```
. gen ageXblack = age*black
. ebalance treat age educ black ageXblack, targets(1)
```

and obtain

```
After:  _webal as the weighting variable
```

		Treat			Control	
	mean	variance	skewness	mean	variance	skewness
age	25.82	51.19	1.115	25.84	82.78	1.156
educ	10.35	4.043	-.7212	10.35	8.053	-.8308
black	.8432	.1329	-1.888	.8421	.133	-1.876
ageXblack	21.91	134.6	-.4435	21.88	161	.01262

Using the `_webal` weights that result from this fit, we can easily verify that the mean of `age` is now balanced in both subgroups of `black`.

```
. bysort black: tabstat age [aweight=_webal], by(treat) s(N me v) nototal

-> black = 0
Summary for variables: age
     by categories of: treat (1 if treated, 0 control)
   treat |         N       mean  variance
---------+----------------------------
       0 |     14816  25.07279  72.12128
       1 |        29  24.93103  40.49507
----------------------------------------
-> black = 1
Summary for variables: age
     by categories of: treat (1 if treated, 0 control)
   treat |         N       mean  variance
---------+----------------------------
       0 |      1176  25.98077  84.71733
       1 |       156  25.98077   53.2835
----------------------------------------
```

Notice that instead of coding the interaction term prior to calling `ebalance`, the user can also call the function using the full functionality for factor variables supported in `Stata` version 11 and higher (see `help fvvarlist` for details). Factor variables can be used to create indicator variables from categorical variables, interactions of indicators of categorical variables, interactions of categorical and continuous variables, and interactions of continuous variables. For example, we obtain similar results as above by coding the interaction term between the continuous variable `educ` and the categorical variable `black` using

```
. ebalance treat age black##c.educ
```

Finally, notice that interactions for a continuous covariate with itself (i.e., a squared term) can also be used to adjust higher order moments for that covariate. For example, coding

```
. ebalance treat age, targets(2)
```

or

```
. gen age2 = age*age
. ebalance treat age age2, targets(1)
```

will both balance the mean and variance of `age`. This is because equality of the means of `age` squared implies the equality of the variances of `age` in the treatment and reweighted control group when the means are also being matched.[4] Notice that a similar approach can be used to adjust higher order moments (e.g., we can adjust the skewness by including similarly generated `age3` or the kurtosis using `age4`.

If the user wants to produce balance figures or tables, he can either use the matrices for the balance results before and after the reweighting that are returned by `e(preBal)` and `e(postBal)` respectively. Alternatively, the user can use the `keep(filename)` option to store the balance results in a `Stata` dataset named `filename.dta` for subsequent summaries (a `replace` option is also available to overwrite an existing `filename.dta` file).

3.5. LaLonde example

We now turn back to the original question of whether preprocessing the data using entropy balancing allows us to more accurately recover the average treatment effect established from the experimental NSW data. To this end, we use `ebalance` and adjust the sample including the means, variances, and skewness of all eleven covariates plus all first order interactions. To do this, we first create all the pairwise interactions.

```
. sysuse cps1re74, clear
. foreach v in age educ black hispan married nodegree re74 re75 u74 u75 {
>   foreach m in age educ black hispan married nodegree re74 re75 u74 u75 {
>     gen `v'X`m'=`v'*`m'
>   }
> }
```

---

[4]The only small difference between these two approaches is that the first coding will adjust to the sample variance computed with the degrees of freedom correction while the second coding will adjust to the sample variance without the degrees of freedom correction. Unless the sample size is very small, the difference between both approaches is negligible.

| | Means | | | Variances | | | Skewness | | |
| | | Controls | | | Controls | | | Controls | |
Covariate	Treated	Pre	Post	Treated	Pre	Post	Treated	Pre	Post
age	25.8	33.2	25.8	51.2	122.0	50.9	1.1	0.3	1.1
educ	10.3	12.0	10.3	4.0	8.2	4.0	−0.7	−0.4	−0.7
black	0.8	0.1	0.8	0.1	0.1	0.1	−1.9	3.3	−1.9
hispan	0.1	0.1	0.1	0.1	0.1	0.1	3.7	3.3	3.7
married	0.2	0.7	0.2	0.2	0.2	0.2	1.6	−0.9	1.6
nodegree	0.7	0.3	0.7	0.2	0.2	0.2	−0.9	0.9	−0.9
re74	2095	14024	2097	23879058	91754832	23782423	3.4	−0.2	3.4
re75	1532	13642	1533	10363576	85747260	10323910	3.8	−0.2	3.8
u74	0.7	0.1	0.7	0.2	0.1	0.2	−0.9	2.3	−0.9
u75	0.6	0.1	0.6	0.2	0.1	0.2	−0.4	2.5	−0.4

Table 1: Covariate balance for raw covariates.

Notice that this includes interactions with the covariate itself, such as `ageXage`, and including these squared terms will adjust the variances of the continuous covariates `age`, `educ`, `re74`, and `re75`. For these continuous covariates, we also include the cubed terms in order to adjust the skewness

```
. foreach v in age educ re74 re75 {
>     gen `v'X`v'X`v' = `v'^3
> }
```

which creates cubed terms such as `ageXageXage`. We then run `ebalance` using the moment restrictions for all the first, second, and third moments as well as first order interactions. Notice that we exclude squared or cubed terms for the binary variables because adjusting the first moment is sufficient to adjust higher moments. We also exclude nonsensical interactions such as for example `blackXhispanic` or `re75Xu75`, etc. Overall we impose 60 moment conditions on the data. The call to `ebalance` is as follows[5]

```
. ebalance treat age educ black hispan married nodegree re74 re75 u74 u75 ///
>    ageXage ageXeduc ageXblack ageXhispan ageXmarried ageXnodegree ///
>    ageXre74 ageXre75 ageXu74 ageXu75 educXeduc educXblack educXhispan ///
>    educXmarried educXnodegree educXre74 educXre75 educXu74 educXu75 ///
>    blackXmarried blackXnodegree blackXre74 blackXre75 blackXu74 ///
>    blackXu75 hispanXmarried hispanXnodegree hispanXre74 hispanXre75 ///
>    hispanXu74 hispanXu75  marriedXnodegree marriedXre74 marriedXre75 ///
>    marriedXu74 marriedXu75  nodegreeXre74 nodegreeXre75 nodegreeXu74 ///
>    nodegreeXu75 re74Xre74 re74Xre75 re74Xu74  re75Xre75 re75Xu74 u74Xu75 ///
>    re75Xre75Xre75 re74Xre74Xre74 ageXageXage educXeducXeduc, keep(baltable)
```

Running the command in 64 bit *Stata* 12 on a desktop computer with an Intel i7 processor with 3.07GHz and 12 GB RAM takes about 2.9 seconds. Table 1 displays the covariate balance on the 1st, 2nd, and 3rd moments for the eleven covariates again before and after entropy balancing (this is an extract from the balance results stored in `baltable.dta` using

---

[5]Notice that we could also use the factor variable commands in *Stata* to code the interactions, but we prefer the explicit coding here for illustration purposes.

the `keep` option). We see that the covariate balance is dramatically improved compared to the unadjusted data. All first order interactions now match as well as all three moments for all eleven covariates.

Does the improved balance move us closer to the experimental target answer? To check this we regress the outcome on the treatment indicator in the reweighted data

```
. svyset [pweight=_webal]
. svy: reg re78 treat


Survey: Linear regression


Number of strata   =          1          Number of obs    =      16177
Number of PSUs     =      16177          Population size  =        370
                                         Design df        =      16176
                                         F(   1,  16176)  =       5.58
                                         Prob > F         =     0.0182
                                         R-squared        =     0.0161


------------------------------------------------------------------------------
             |               Linearized
        re78 |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       treat |   1761.344   745.5099     2.36   0.018     300.062    3222.626
       _cons |     4587.8   472.2286     9.72   0.000    3662.179     5513.42
------------------------------------------------------------------------------
```

We find a treatment effect estimate of USD 1,761 which suggests that the entropy balancing preprocessing step moves us very close to the experimental target answer. The estimate is also fairly efficient with a confidence interval that ranges from USD $[300, 3223]$ (notice that this treats the weights as fixed).

3.6. Survey reweighting

Apart from the two-group setup with a treatment and a control group, `ebalance` can also be used to reweight a single sample to a set of known target moments. This scenario often occurs in survey analysis, where a sample should be reweighted to some known features of the target population. To accomplish this task, the researcher can use the `manualtargets(numlist)` option in the `ebalance` command to specify values for a set of target moments that correspond to the variables in the `covar` list. Notice that no `treat` variable should be specified in this case since there is only a single group. For example, imagine the data constitutes a single data sample, and the user likes to reweight this sample such that the means of the variables `age`, `educ`, `black`, and `hispan` match the values 28, 10, .1, and .1 respectively. We call

```
. ebalance age educ black hispan, manualtargets(28 10 0.1 0.1)
```

and obtain

```
Data Setup
Covariate adjustment: age educ black hispan

Optimizing...
Iteration 1: Max Difference = 53608.6
.
Iteration 14: Max Difference = .000669546
maximum difference smaller than the tolerance level; convergence achieved

No. of units adjusted: 16177 total of weights: 16177

Before: without weighting

          |      mean    variance   skewness
----------+-------------------------------
      age |     33.14      121.8        .358
     educ |     12.01      8.225      -.4146
    black |    .08234     .07556       3.039
   hispan |    .07189     .06673       3.315

After:  _webal as the weighting variable

          |      mean    variance   skewness
----------+-------------------------------
      age |        28      107.9       .8476
     educ |        10       11.1      -.9571
    black |    .09999        .09       2.667
   hispan |        .1        .09       2.667
```

so the reweighted sample now matches the desired target moments. Notice that the manual option is not compatible with the `targets` option, but otherwise the command works similar to the two-group case discussed above. The fitted weights are stored in the `_webal` variable.

3.7. Further options and issues

Apart from the functionality described above, `ebalance` offers a few additional options that can be useful to handle special cases. In this section we briefly discuss these extra options and also elaborate on some further issues to keep in mind when using the package. Additional details for all options can be found in the help file by typing `help ebalance` at the *Stata* command prompt.

Base weights

The `basewt(varname)` option offers users the opportunity to supply their own base weights for the entropy balancing step (in lieu of the default weights which are uniformly distributed). If this option is specified, then `ebalance` will start the optimization from a set of user specified base weights supplied in the variable from the `basewt(varname)` option. This can be helpful in cases where the researcher already has an initially estimated propensity score weight that

she likes to "overhaul" with entropy balancing by imposing exact balance constraints. Notice that the user specified base weights are only applied for the control units, unless the option `wttreat` is also specified in which case the base weights are also applied to the treated units. This can be useful in situations where the researcher has some existing survey weights that need to be accounted for in computing the moment conditions from the treated units.

Normalization constant

As can be seen in the `ebalance` output above, the function by default normalizes the control group weights such that they add up to the number of treated units. However, this normalizing constant is of course arbitrary and can be reset to other values if needed. The `normconst(real)` option allows the user to change the normalizing constant by specifying a number for the ratio of the sum of weights for the treated units to the sum of weights for the control units (see help file for details). The default is a ratio of one. Alternatively, the researcher can also re-scale the weights stored in `_webal` post hoc.

Optimization settings

The options `maxiter(integer)` and `tolerance(real)` control two settings of the optimization algorithm. `maxiter(integer)` allows the user to set the maximum number of iterations (the default is 20) and the `tolerance(real)` option allows users to change the tolerance criteria that is used to declare convergence in the optimization (the default is .015). The tolerance number refers to the maximum deviation from the moment conditions across all the variables included in `covars`. The user can lower the tolerance level to obtain stricter balance (i.e., exact to a certain number of digits) or loosen it to allow for some small deviations. Notice that if `ebalance` does not achieve a level of covariate balance that is within the specified tolerance level in the maximum number of iterations, it still returns the results from the balance obtained in the last iteration.

Caveat about constraint specification

Finally, it is important to keep in mind that the method – just like any other – provides no panacea for achieving covariate balance. As described in Hainmueller (2012) the user has to be careful not to impose unrealistic or even inconsistent balance constraints. For example, it makes no sense to specify balance constraints that imply that a control group should be reweighted to have 20% women and 20% males. Similar, it is unrealistic to reweight a control group with 10% women to one with 90% women; if the two groups are radically different than there is not much information in the data to identify the counterfactual of interest. Similarly, the researcher cannot impose more balance conditions than control group observations and if too many balance conditions are included with limited data, the constraint matrix may be close to singular and the entropy balancing algorithm may break down. When convergence is not achieved, `ebalance` displays the most demanding moment constraint. In such cases the user needs to reduce the number of constraints or gather more data. By default, `ebalance` also computes a check for the overlap in the covariate distributions and alerts users in the cases where the target moments are outside of the range of the covariate distributions.

4. Conclusion

In this article we have described how to implement entropy balancing using the **ebalance** package for `Stata`. The method allows researchers to create balanced samples for observational studies with binary treatments or to reweight a dataset to some known target moments. We illustrated the use of the `ebalance` function using various examples from the LaLonde data.

Future work may consider how entropy balancing could be combined with other matching methods that are implemented in `Stata` such as **nnmatch** (Abadie, Drukker, Herr, and Imbens 2004), **psmatch2** (Leuven and Sianesi 2003), or **cem** (Iacus, King, and Porro 2009). As discussed in Hainmueller (2012), researchers could for example first run a coarsened exact matching to discard extreme control and or treated units and then follow up with entropy balancing in the reweighted data to further balance out the covariates. Similar, entropy balancing can be combined with regression approaches where the user first reweights the data by adjusting for the covariates that are predictive of the treatment, and then applies the weights to a regression model that aims to model the relationship between the outcome, treatment, and additional covariates that are predictive of the outcome. This procedure would be akin to doubly robust regression (Robins, Rotnitzky, and Zhao 1995; Hirano and Imbens 2001) and can further help to reduce model dependency.

Finally, some extensions to `ebalance` are currently under development. In particular, we consider implementing a procedure to refine the entropy balancing weights by trimming large weights to lower the variance of the weights and thus the variance for the subsequent analysis.

Acknowledgments

References

Abadie A, Drukker D, Herr JL, Imbens GW (2004). "Implementing Matching Estimators for Average Treatment Effects in `Stata`." *Stata Journal*, **4**(3), 290–311.

Abadie A, Imbens G (2011). "Bias Corrected Matching Estimators for Average Treatment Effects." *Journal of Business and Economic Statistics*, **29**(1), 1–11.

Dehejia R, Wahba S (1999). "Causal Effects in Nonexperimental Studies: Reevaluating the Evaluation of Training Programs." *Journal of the American Statistical Association*, **94**, 1053–1062.

Deming WE, Stephan FF (1940). "On the Least Squares Adjustment of a Sampled Frequency Table When the Expected Marginal Totals Are Known." *Annals of Mathematical Statistics*, **1940**, 427–444.

Diamond AJ, Sekhon J (2006). "Genetic Matching for Causal Effects: A General Multivariate Matching Method for Achieving Balance in Observational Studies." Working Paper.

Erlander S (1977). "Entropy in Linear Programs – An Approach to Planning." *Technical Report LiTH-MAT-R-77-3*, Department of Mathematics, Linkrping University.

Hainmueller J (2012). "Entropy Balancing for Causal Effects: A Multivariate Reweighting Method to Produce Balanced Samples in Observational Studies." *Political Analysis*, **20**(1), 25–46.

Hainmueller J (2013). **ebal**: *Entropy Reweighting to Create Balanced Samples*. R package version 0.1-4, URL http://CRAN.R-project.org/package=ebal.

Hirano K, Imbens G (2001). "Estimation of Causal Effects Using Propensity Score Weighting: An Application of Data on Right Hear Catherization." *Health Services and Outcomes Research Methodology*, **2**, 259–278.

Hirano K, Imbens G, Ridder G (2003). "Efficient Estimation of Average Treatment Effects Using the Estimated Propensity Score." *Econometrica*, **71**(4), 1161–1189.

Ho DE, Imai K, King G, Stuart EA (2007). "Matching as Nonparametric Preprocessing for Reducing Model Dependence in Parametric Causal Inference." *Political Analysis*, **15**(3), 199.

Iacus S, King G, Porro G (2012). "Causal Inference without Balance Checking: Coarsened Exact Matching." *Political Analysis*, **20**(1), 1–24.

Iacus SM, King G, Porro G (2009). "**cem**: Software for Coarsened Exact Matching." *Journal of Statistical Software*, **30**(9), 1–27. URL http://www.jstatsoft.org/v30/i09/.

Imbens GW (2004). "Nonparametric Estimation of Average Treatment Effects under Exogeneity: A Review." *Review of Economics and Statistics*, **86**(1), 4–29.

Ireland CT, Kullback S (1968). "Contingency Tables with Given Marginals." *Biometrika*, **55**, 179–188.

LaLonde RJ (1986). "Evaluating the Econometric Evaluations of Training Programs with Experimental Data." *American Economic Review*, **76**, 604–620.

Leuven E, Sianesi B (2003). "**psmatch2**: Stata Module to Perform Full Mahalanobis and Propensity Score Matching, Common Support Graphing, and Covariate Imbalance Testing." Statistical Software Components, Boston College Department of Economics. URL http://ideas.repec.org/c/boc/bocode/s432001.html.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Robins JM, Rotnitzky A, Zhao LP (1995). "Analysis of Semiparametric Regression Models for Repeated Outcomes in the Presence of Missing Data." *Journal of the American Statistical Association*, **90**(429).

Rosenbaum PR, Rubin DB (1983). "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika*, **70**(1), 41–55.

Rubin DB (2006). *Matched Sampling for Causal Effects*. Cambridge University Press.

Särndal CE, Lundström S (2006). *Estimation in Surveys with Nonresponse*. John Wiley &
    Sons.

Sekhon JS (2009). "Opiates for the Matches: Matching Methods for Causal Inference." *Annual
    Review of Political Science*, **12**, 487–508.

StataCorp (2011). *Stata Data Analysis Statistical Software: Release 12*. StataCorp LP, College
    Station, TX. URL http://www.stata.com/.

Zaslavsky A (1988). "Representing Local Reweighting Area Adjustments by of Households."
    Survey Methodology, **14**(2), 265–288.

Affiliation:

Jens Hainmueller
Department of Political Science
Massachusetts Institute of Technology
Cambridge, MA 02139, United States of America
E-mail: jhainm@mit.edu
URL: http://www.mit.edu/~jhainm/