PRE-PROCESSING HOUSING DATA VIA MATCHING: SINGLE MARKET

KLAUS MOELTNER

1. Introduction

In this script we show, for a single housing market, how mis-specification of the hedonic price function can lead to biased estimates of treatment effects. We also illustrate how pre-processing the data via matching can alleviate mis-specification problems.

This is because matching breaks the dependence between treatment and other observable variables. This, in turn, makes it less important how these other variables are controlled for in the hedonic price function.

2. Data generation

We create a simulated price function for treated (ex.: "in floodzone A") and untreated (ex.: "in floodzone X") properties, using summary statistics from our actual data set as guidance (data set "Target" in STATA, using homes in A-zones as treated, and homes in X-zones as controls).

We generate three explanatory variables: square footage, bedrooms, and bathrooms. All three follow closely their distribution in the actual data.

```
R> nT<-500 #number of treated
R> nC<-5000#number of controls
R> n<-nT+nC
R> norig<-n
R> sqftT<-rtruncnorm(nT,a=600,b=15000,mean=2800,sd=1600)/1000 #in 1000s
R> sqftC<-rtruncnorm(nC,a=300,b=23000,mean=3600,sd=1700)/1000
R> #draw n observations for this truncated normal density and place into an nx1 vector
R> #to see summary stats, type summary(sqftT) in the Console
R> pbedT<-c(0.0444, 0.2611, 0.5098, 0.1528, 0.0249, 0.007)
R> pbedC<-c(0.0136, 0.1458, 0.5243, 0.2764, 0.0332, 0.0067)
R> bedrT<-sample(1:6,nT,replace=TRUE,prob=pbedT)
R> bedrC<-sample(1:6,nC,replace=TRUE,prob=pbedC)
R> #
R> pbathT<-c(0.5507,0.3552,0.0782,0.0159)
R> pbathC<-c(0.3931,0.4555,0.1272,0.0242)
R> bathrT<-sample(1:4,nT,replace=TRUE,prob=pbathT)
R> bathrC<-sample(1:4,nC,replace=TRUE,prob=pbathC)
```

Now generate the dependent variable using the "true" hedonic price function, which includes a quadratic for sqft (in units of 1000) and an interaction between bedrooms and bathrooms.

```
R> sqft<-c(sqftT,sqftC) #stick both samples into one long vector
R> sqft2<-sqft^2
```

```
R> beds<-c(bedrT,bedrC)
R> baths<-c(bathrT,bathrC)
R> bbint<-beds*baths
R> treat<-matrix(c(rep(1,nT),rep(0,nC)))
R> X<-cbind(rep(1,n),sqft,sqft2,beds,baths,bbint,treat)
R> #
R> bvec=c(50,75,-0.5,10,15,5,-25)
R> eps<-rnorm(n,0,25)
R> y<-X %*% bvec+eps #sales price in 000s
R> yorig<-y
R> yT<-y[1:nT]
R> yC<-y[(nT+1):n]
```

Let's summarize the data graphically and in a table, for treated, controls, and all observations.

TABLE 1. summary statistics, treated

variable	mean	std	min	max
sqft	3.12	1.40	0.62	8.90
bedrooms	2.87	0.85	1.00	6.00
bathrooms	1.58	0.71	1.00	4.00
price ($000s)	328.66	103.67	80.90	756.68

TABLE 2. summary statistics, controls

variable	mean	std	min	max
sqft	3.76	1.60	0.32	10.11
bedrooms	3.20	0.79	1.00	6.00
bathrooms	1.77	0.75	1.00	4.00
price ($000s)	410.24	119.18	96.71	833.86

3. Checking for balance

Let's compare the distributions of treated and controls using empirical quantile-quantile (eQQ) - plots. This is the preferred method of Ho, Imai, King and Stuart (2007) for a "visual balance check." This has to be done variable-by-variable. The plot breaks the empirical distribution of a given variable for treated and controls into equal empirical quantiles, and plots the resulting values of one group against those of the other.

Specifically, for 2 vectors of equal length, the "qqplot" function in R simply produces a pairwise plot of all points of the sorted data. For vectors of unequal length, it reduces the (sorted) longer vector to the same number of points as the shorter vector, using interpolation, then plots the pairs.

This really only works for continuous variables, so here square footage. We will add a QQ plot for sales price as well, not as a balance check, but simply for comparison. For the remaining integer variables, a grouped bar chart is probably more meaningful. We can see that the controls have,

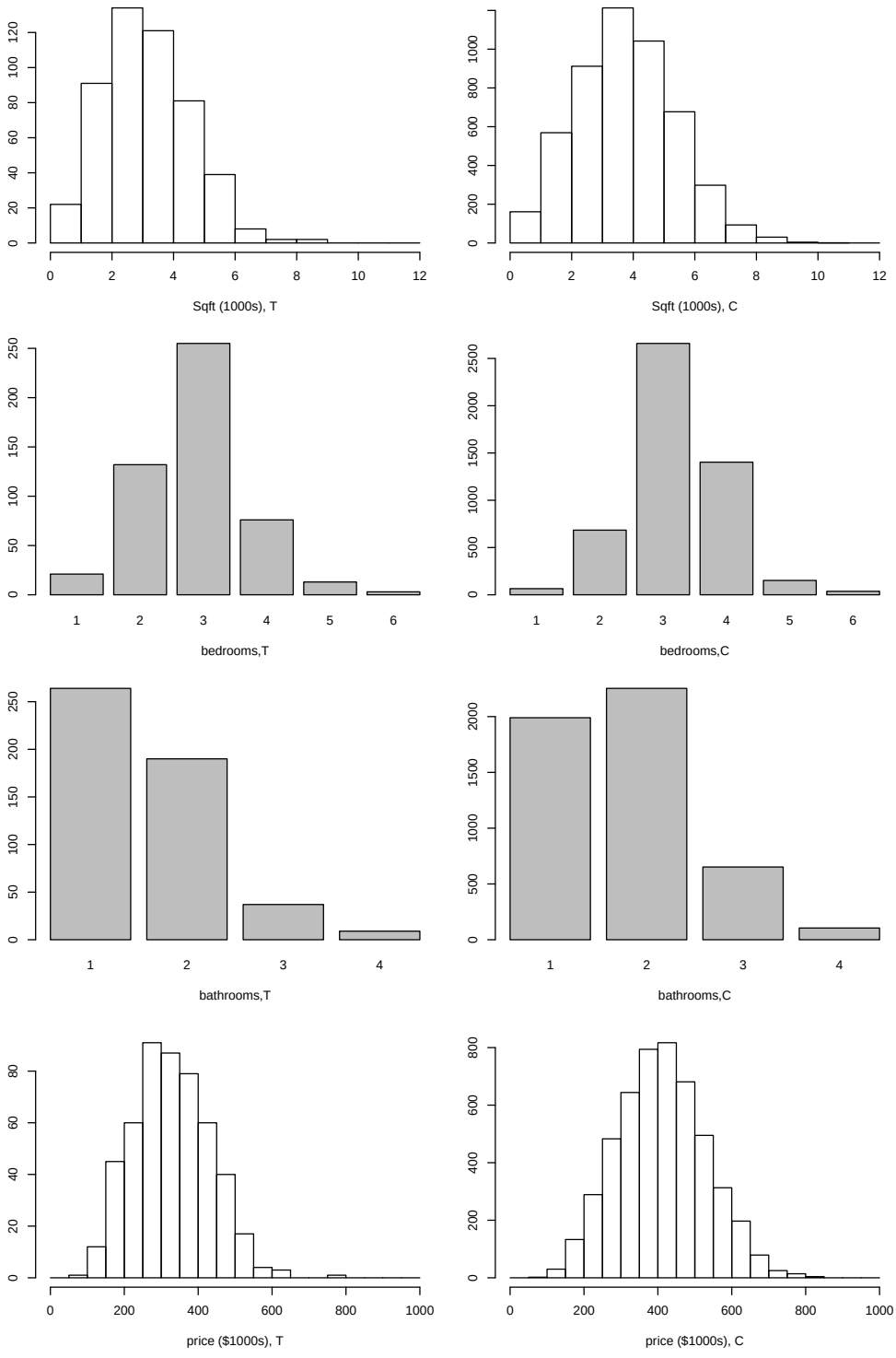

FIGURE 1. Sample statistics

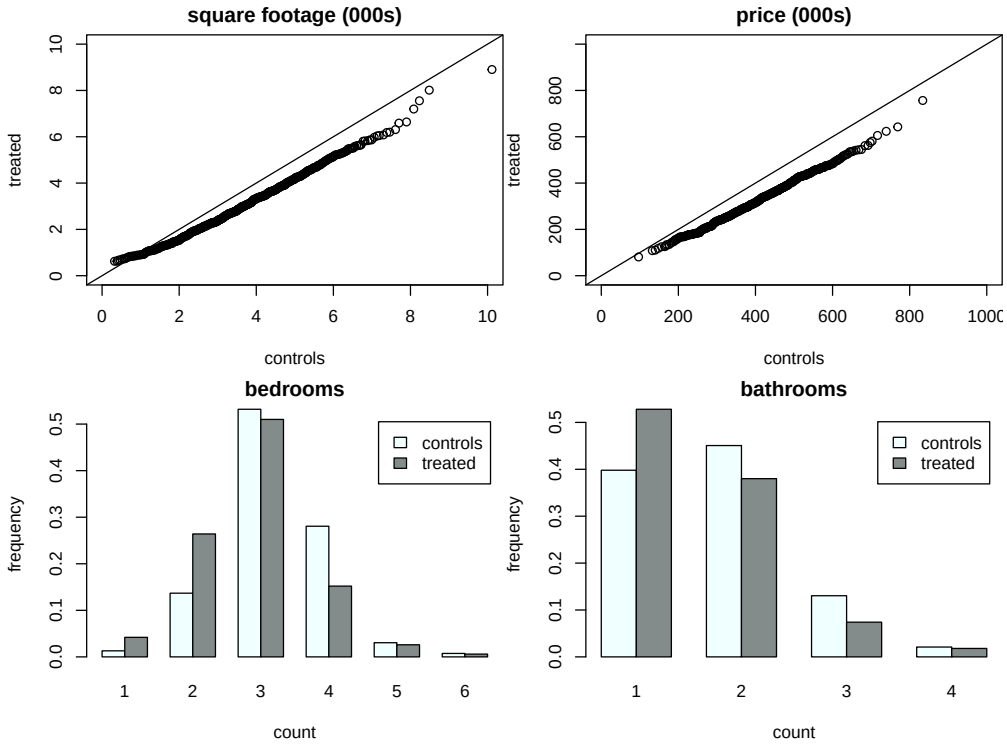

FIGURE 2. QQ and frequency plots for treated vs. controls

overall, larger properties and more expensive homes (percentiles lie below those of the treated). There are also higher proportions of controls at the upper end of the bedroom and bathroom distribution. Our goal with matching will be to get the QQ plot as close to the diagonal line as possible.

Sekhon (2011) also uses QQ plots to motivate his genetic search algorithm for balance. A submodule of his "*Matching*" program, called "*qqstats*" also generates numerical QQ statistics, such as mean, median, and maximum deviation for equal quantile points.[1] We will report these in addition to standardized differences in means. These are the raw differences in means for a given variable, divided either by the standard deviation of the treated sample (as used in Ho, Imai, King and Stuart (2011)) or the weighted standard deviation for the whole sample. The latter produces what Imbens and Rubin (forthcoming) call *normalized differences*.[2]

If the ATT is of primary interest, it makes more sense to normalize by the standard deviation for the treated. If the ATE is required, normalizing by the weighted full standard deviation is probably more meaningful. Our focus is on ATT, so we'll use the former. As can be seen from the table,

normalized distances between means are substantial for all explanatory variables. The same holds for mean QQ differences. For example, for sqft, a point on the QQ plot is, on average, 610 square

[1]See also Ho et al. (2007) (ftn.16).

[2]These authors refer to balance as "overlap." They suggest that any normalized differences above 0.25 are worrisome.

Table 3. balance statistics

variable	norm.diff	mean QQ diff	med QQ diff	max QQ diff
price	-0.787	81.609	82.971	129.843
sqft	-0.456	0.647	0.673	1.276
beds	-0.383	0.326	0.000	1.000
baths	-0.272	0.194	0.000	1.000

feet away form the diagonal line (in fact, below the diagonal line, as we can see from the actual plot.)

4. Analyzing the unbalanced data

Let's estimate a few models (including the correct one) via OLS, using the raw data set. We will try five models:

(1) M1: Simple linear (basic)
(2) M2: Basic with squared term for square footage
(3) M3: Basic with interaction of bedrooms, bathrooms
(4) M4: Basic with squared term for square footage and interaction of bedrooms, bathrooms (correct model)
(5) M5: As M3, but using log(sqft)

We evaluate the performance of each model following Diamond and Sekhon (2013), using bias (in percent) for the treatment estimate, and root mean squared error (RMSE) for overall model fit. As the following table shows, our linear models 1-4 perform quite well despite mis-specification (for models 1-3). However, the falsely specified non-linear model in square footage (M5) produces a very misleading treatment estimate, with bias in excess of 30%. Not surprisingly, the correct model (M4) generates the smallest bias and lowest RMSE.

Table 4. Treatment estimate, raw data

model	true	est.	bias(%)	RMSE
M1	-25.000	-24.604	-1.584	25.039
M2	-25.000	-24.593	-1.629	24.992
M3	-25.000	-25.042	0.169	24.789
M4	-25.000	-25.029	0.116	24.744
M5	-25.000	-32.719	30.875	44.405

5. Pre-processing the data with matching

Now let's narrow the set of controls to achieve better balance in the final data set used for analysis. We will use the genetic search algorithm proposed by Diamond and Sekhon (2013) and described in detail in Sekhon (2011) to select a set of controls that produce empirical distributions for the observed explanatory variables (sqft, beds, baths) that are as similar as possible to those of the treated units.

The Diamond and Sekhon (2013) algorithm, henceforth referred to as *GenMatch* is based on a weighted Mahalanobis matrix as distance criterion for matching. The optimization routine then looks for the vector of weights that produces "optimal balance," as measured by a variety of test statistics. In math notation:

$$d_{ij} = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)' \mathbf{V}^{-1/2} \mathbf{W} \mathbf{V}^{-1/2} (\mathbf{x}_i - \mathbf{x}_j)} \tag{1}$$

where d_{ij} is the vector distance between treated observation i and control j, $\mathbf{x}_i$ and $\mathbf{x}_j$ are corresponding vectors of observables, $\mathbf{V}$ is the variance-covariance matrix of these observables in the entire data, and $\mathbf{W}$ is a diagonal matrix with weights on the diagonal. The *GenMatch* routine optimizes over these weights - not to minimize individual, average, or total distance, but to achieve optimal balance as described above.

This process requires matching at each iteration to check on balance. By default, this matching is based on the nearest-neighbor method (with replacement, and keeping ties) proposed in Abadie and Imbens (2006). Conveniently, the *GenMatch* algorithm can be run in parallel on multi-core PCs or networks. This substantially cuts down on computation time.

Sekhon (2011) distinguishes between matrix $\mathbf{X}_M$, which includes the variables that need to be matched, and $\mathbf{X}_B$, which includes the variables to be used for balance checks. He recommends adding higher order terms and interactions to the latter. He also suggests to add the propensity score (PS) $\pi(\mathbf{Z})$ to the matching matrix $\mathbf{X}$. To be specific, he prefers using the linear prediction from the PS equation, not the predicted probabilities to avoid boundary problems near 0 and 1. In addition, he warns against over-specifying the propensity score equation. A simple logit with the remaining variables in $\mathbf{X}_M$ should work fine - that's what he uses in all of his examples.

The propensity score is the probability of receiving treatment, given observed variables $\mathbf{Z}$. As discussed in Ho et al. (2007), it is a *balancing score*. This means that if the treated and controls have the same distribution for $\pi(\mathbf{Z})$ (i.e. are "balanced on the propensity score"), they will automatically be balanced on all observables that enter the PS equation as well. However, this only holds for the true PS, which is unknown. So using the PS alone as guidance for matching is generally not a good idea, even with a large sample. However, the empirical PS is still a usefull tool to to guide the *GenMatch* algorithm towards optimal balance. We will therefore follow Sekhon's advice and include the PS in our matching matrix $\mathbf{X}_M$.

With this data set and the settings described in the code chunk, *GenMatch* runs for about 2.5 minutes on a cluster of 8 local cores. It goes through 9 "generations" of weight batches. The primary output of *GenMatch* is the diagonal of the weight matrix $\mathbf{W}$ in equation (1) above. These values are captured in the following table:

Table 5. Optimal weights from GenMatch

variable	weight
sqft	1.226
beds	152.506
baths	184.867
linPS	873.558

We now need to run the *Match* function to obtain the final matched sample and a non-parametric estimate for the ATT (if desired). This function uses the weights produced by *GenMatch* shoen above. Naturally, we'll use the same general settings for *Match* that we used for *GenMatch*, here matching with replacement and allowing for ties.

Note that we will run this final matching procedure using Abadie, Drucker, Herr and Imbens (2002)'s bias adjustment, which corrects the purely nonparametric estimate of the price differential of a treated relative to its control(s) with predictions from a secondary regression model *run only on the matched controls*.

From the output of *Match* we can tell that the matched data set has 2001 observations. Since we asked for 4 best matches for each of our 500 treated, this implies that there were no ties - each treated is matched with exactly 4 controls. However, since we started with 5000 controls, clearly many controls were discarded. Specifically, we ended up using 1502 unique control observations. This implies that the matched data includes 499 replicated controls - controls that were used for more than one treated.

Let's see if balance has improved based on our graphical inspection and our quantitative measures from above.

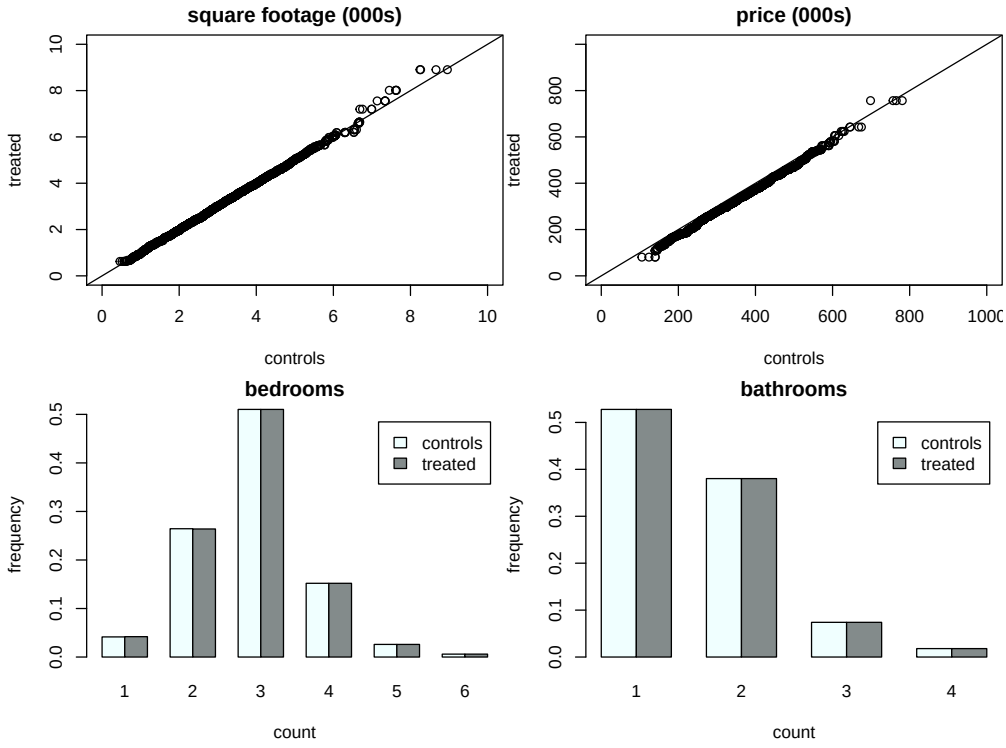

FIGURE 3. QQ and frequency plots for treated vs. controls

TABLE 6. balance statistics

variable	norm.diff	mean QQ diff	med QQ diff	max QQ diff
price	-0.240	24.943	25.422	58.972
sqft	-0.001	0.018	0.010	0.657
beds	-0.001	0.000	0.000	1.000
baths	0.000	0.000	0.000	0.000

As can be seen from both the graphs and table, balance has improved dramatically for all variables.

Once we have this final sample, we can two things: (1) Run again our regression models on the (weighted) matched data and estimate a single treatment effect parametrically, as before, or (2) directly estimate the ATT non-parametrically with or without regression adjustment, a la Abadie et al. (2002). The latter approach generally uses the data more efficiently, as it pays attention to the actually matched pairs. It also produces a separate ATT for each treated home (which are then averaged over homes). Getting individual treatment effects can be useful down the line.[3]

However, the main purpose of this note is to show that using OLS on the matched data is more robust to model mis-specification, so we will re-run or original models.

6. Analyzing the balanced data

We run the same 5 models as before, using all 500 treated observations, but only the 1502 unique controls coming out of the matching process. Each of these is weighted depending on how often they were used in the matched sample. Ho et al. (2011), section 5.3, explain the computation of these weights. The Sweave code underlying this documents shows the exact programming steps.

We use the same criteria as above (relative bias of treatment estimate, and overall fit via RMSE) to assess model quality.

TABLE 7. Treatment estimate, M1

model	true	est.	bias(%)	RMSE
M1	-25.000	-25.130	0.522	25.755
M2	-25.000	-25.101	0.402	25.748
M3	-25.000	-25.187	0.747	25.572
M4	-25.000	-25.155	0.619	25.563
M5	-25.000	-24.981	-0.077	37.974

As is evident from the new set of results, matching has greatly improved the performance of the fatally flawed model 5, which, coincidentially, now shows the smallest bias. Bias is also decreased for the first two models relative to the original OLS results.
It is quite obvious that the estimated treatment effect has become almost perfectly robust to model specification, with all bias figures lying below the single percentage mark.

[3]In passing, we note that the non-parametric, bias-adjusted ATT is -24.744, with a heteroskedasticiy-robust standard error of 1.304

References

Abadie, A., Drucker, D., Herr, J. and Imbens, G. (2002). Simple and bias-corrected matching estimators for average treatment effects. NBER Technical Working paper 283.

Abadie, A. and Imbens, G. (2006). Large-sample properties of matching estimators for average treatment effects, *Econometrica* **74**: 235–267.

Diamond, A. and Sekhon, J. (2013). Genetic matching for estimating causal effects: A general multivariate matching method for achieving balance in observational studies, *The Review of Economics and Statistics* **95**: 932–945.

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

Ho, D., Imai, K., King, G. and Stuart, E. (2011). MATCHIT: Nonparametric preprocessing for parametric causal inference. user manual posted at: `http://r.iq.harvard.edu/docs/matchit/2.4-20/matchit.pdf`; last accessed Jan. 20, 2015.

Imbens, I. and Rubin, D. (forthcoming). *Causal Inference in Statistics and the Social Sciences*, Cambridge University Press.

Sekhon, J. (2011). Multivariate and propensity score matching software with automated balance optimization: The Matching package for R, *Journal of Statistical Software* **42**: 1–52.

Department of Agricultural and Applied Economics, Virginia Tech
E-mail address: `moeltner@vt.edu`