Motivation: Confounders and Control Variables

Probably most empirical economics papers are interested in estimating a causal effect $\alpha$ of some explanatory variable $d$ on an dependent variable $y$. (Note that $d$ is not necessarily a dummy, I just stick to prevalent notation in the literature). If our data does not come from a randomized experiment this is generally a hard task because there are essentially always some confounders. Unless we appropriately control for confounders (or circumvent them e.g. by instrumental variable estimation), we get an inconsistent estimator $\hat \alpha$ of the causal effect.

Consider the following simple example where our OLS estimator $\hat \alpha$ has a positive bias if we don't control for the confounder `xc`:

```
n = 10000
xc = rnorm(n,0,1) # confounder
d = xc + rnorm(n,0,1)
alpha = 1; beta = 1;
y = alpha*d + beta * xc + rnorm(n, 0,1)
# OLS regression without controlling for confounder
coef(lm(y~d))[2]
```

```
##        d
## 1.485069
```

If we add the confounder as control variable, we get an unbiased OLS estimator:

```
coef(lm(y~d+xc))[2]
```

```
##         d
## 0.9987413
```

In empirical studies, the main problem is typically that we just don't have data for all relevant confounders. However, there are some applications where we don't have too many observations but a lot of potential control variables, in particular if we also consider non-linear effects and interaction terms. An example can be found in the vignette (Chernozhukov, Hansen and Spindler, 2016) of the hdm package (which implements the post-double-selection from Belloni et. al.). The example is based on the empirical growth literature, which tries to estimate the causal effect of some variable, like initial GDP, on GDP growth. Gross country panel data sets are not very large, but we can observe a lot of variables for a country. If one is optimistic, one may believe that some subset of the potential control variables is able to effectively control for all confounders.

But if we don't have enough observations, we need some method to select the relevant control variables.

Framework of Monte Carlo Simulation

Let us study different approaches for variable selection using a Monte-Carlo simulation with the following data generating process for $y$ and $d$. (If Mathjax is not well rendered on a blog aggregator click here.):

$$y = \alpha d + \sum_{k=1}^{K_c} \beta^{cy}_k {x^c_k} + \sum_{k=1}^{K_y} \beta^y_k {x^y_k} + \varepsilon^y$$

$$d = \sum_{k=1}^{K_c} \beta^{cd}_k {x^c_k} + \sum_{k=1}^{K_e} \beta^e_k {x^e_k} + \varepsilon^d$$

We have the following potential control variables that are all independently normally distributed from each other:

- $x^c_k$ is one of $K_c$ confounders that directly affect both $d$ and $y$. For a consistent OLS estimation of $\alpha$, we need to control for all confounders.
- $x^y_k$ is one of $K_y$ variables that only affect the dependent variable $y$ but not the explanatory variable $d$. Whether we would add it or not in an OLS regression should not affect the bias of our estimator $\hat \alpha$.

- $x^e_k$ is one of $K_e$ variables that only affect $d$ but not through any other channel the dependent variable $y$. It constitutes a *source of exogenous variation* in $d$. We can estimate in an OLS regression $\alpha$ more precisely if we don't add any $x^e_k$ to the regression. Also, we will see that adding fewer $x^e_k$ can reduce the bias of an OLS estimator that arises if we have not perfectly controlled for all confounders.

- We also observe $K_u$ variables $x^u_k$ that neither affect $y$ nor $d$. They are just uncorrelated noise variables that we ideally leave out of our regressions.

In this Github repository you can find scripts that contains the function `lasso_sim` and other utility functions that help us to run the simulation.

The following code simulates a data set with $n=1000$ observations, $K_c=50$ confounders, $K_y=50$ variables that affect only $y$, just a single observed variable $x^e_k$ that provides a source of exogenous variation and $K_u=1000$ explanatory variables that are uncorrelated with everything else. The causal effect of interest $\alpha$, as well as all other regression coefficients and standard deviations are equal to 1.

```
source("lasso_tools.R")
source("lasso_sim.R")
set.seed(1)
mat = lasso_sim(alpha=1, n=1000,Kc=50,Ke=1,Ky=50,Ku=1000,return.what = "data")
dim(mat) # more variables than observations
```

```
## [1] 1000 1103
```

```
mat[1:3,1:5] # show excerpt
```

```
##              y         d        xc1        xc2        xc3
## [1,] -1.319487 -3.570383 -0.6264538  1.1349651 -0.8861496
## [2,] 15.526450  5.809369  0.1836433  1.1119318 -1.9222549
## [3,] 21.525964  1.011645 -0.8356286 -0.8707776  1.6197007
```

Let us first estimate three regressions via OLS:

```
y = mat[,1]
X = cbind(1,mat[,-1]); colnames(X)[1] = "const"

# Using all x does give non-sensible result
# because we have too many variables
coef(lm.fit(y=y,x=X))[2]
```

```
##        d
## 3.112387
```

```
# Short regression y ~ d yields positive bias
# due to omitted confounders
coef(lm.fit(y=y,x=X[,1:2]))[2]
```

```
##       d
## 1.94576
```

```
# Controlling for all confounders eliminates bias
coef(lm.fit(y=y,x=X[,1:52]))[2]
```

```
##        d
## 1.015087
```

If we know which variables are the confounders, we can easily consistently estimate the causal effect $\alpha=1$. But let's assume we don't know which of the 1101 potential control variables are the confounders.

Lasso, Post-lasso and Post Double Selection

From the machine learning toolbox, lasso regressions seem natural candidates to select relevant control variables. Consider a linear regression with $K$ standardized explanatory variables with corresponding estimators and residuals denoted by $\hat \beta$ and $\hat \varepsilon(\hat \beta)$, respectively. The lasso estimator solves the following optimization problem:

\[\min_{\hat \beta} \sum_{i=1}^n {\hat \varepsilon_i(\hat \beta)^2} + \lambda \sum_{k=1}^K {|\hat \beta_k|}\]

The first term is just the sum of squared residuals, which the OLS estimator minimizes. The second term penalizes larger absolute values of the estimated coefficients. The penalty parameter $\lambda \gt 0$ will be chosen in an outer loop e.g. by cross-validation or using a criterion like the corrected AIC. Lasso estimates typically have many coefficients $\hat \beta_k$ equal to zero. In this sense the lasso estimator selects a subset of explanatory variables whose estimated coefficients are non-zero.

But also the coefficients of the selected variables will be typically attenuated towards 0 because of the penalty term. The *post-lasso* estimator avoids this attenuation by simply performing an OLS estimation using all the selected variables from the lasso estimation.

The following code estimates a lasso and post-lasso regression for our dependent variable $y$ in our simulated data set and examines the corresponding estimator $\hat \alpha$ of our causal effect of interest. I use the package gamlr developed by Matt Taddy. He uses it extensively in his great new text book Business Data Science. It discusses a lot of modern econometric developments at the intersection with machine learning in a very intuitive and applied fashion.

The `gamlr` function uses by default the corrected Akaike Information Criterion to select the penalty parameter $\lambda$. This avoids the random fluctuation from cross validation (used in the popular cv.glmnet function) and makes the `gamlr` function blazingly fast.

```
# Lasso estimator
library(gamlr)
lasso = gamlr(x=X,y=y)
# lasso_coef is a helper function in lasso_tools.R
# because coef(lasso) returns a sparse matrix
coefs = lasso_coef(lasso,keep.intercept = FALSE)
coefs[1] # still biased
```

```
##        d
## 1.955174
```

```
# Post lasso estimator
vars = names(coefs)
post.lasso = lm.fit(y=y,x=X[,c("const",vars)])
coef(post.lasso)[2] # post-lasso estimator also still biased
```

```
##        d
## 1.979171
```

We see that both the lasso and post-lasso estimators of the causal effect $\alpha$ are still biased. Both are roughly the same size as the estimate in the short OLS regression of $y$ on just $d$. To understand why, let’s have a look at the selected variables:

```
vars
```

```
##  [1] "d"      "xc3"    "xc39"   "xe1"    "xu314"  "xu700"  "xu734"  "xu763"  "xu779"
## [10] "xu831"  "xu870"  "xy1"    "xy2"    "xy3"    "xy4"    "xy5"    "xy6"    "xy7"
## [19] "xy8"    "xy9"    "xy10"   "xy11"   "xy12"   "xy13"   "xy14"   "xy15"   "xy16"
## [28] "xy17"   "xy18"   "xy19"   "xy20"   "xy21"   "xy22"   "xy23"   "xy24"   "xy25"
## [37] "xy26"   "xy27"   "xy28"   "xy29"   "xy30"   "xy31"   "xy32"   "xy33"   "xy34"
## [46] "xy35"   "xy36"   "xy37"   "xy38"   "xy39"   "xy40"   "xy41"   "xy42"   "xy43"
## [55] "xy44"   "xy45"   "xy46"   "xy47"   "xy48"   "xy49"   "xy50"
```

```
# A helper function to count selected variables by type
```

```
vars_counts(vars)
```

```
## num.vars       xc       xe       xu       xy
##       60        2        1        7       50
```

While the lasso regression selects all 50 `xy` variables that only affect $y$, it only selects 2 of the 50 confounders `xc`. So 48 confounders remain uncontrolled and create massive bias. This quite asymmetric selection may be a bit surprising given that in our simulation the confounders `xc` affect `y` as strongly as the `xy` variables ($\beta_k^{cy} = \beta_k^{y} = 1$). My intuition is that our (also selected) variable of interest `d` already captures some effect of the confounders on `y`. Thus the confounders are less important to predict `y` and are therefore not selected.

The *post double selection* method by Belloni et. al. (2014) selects the control variables as follows. We run two lasso regressions. The first regresses `d` on all potential controls. The second regresses `y` on all potential controls (excluding `d`). Then we use the union of the selected variables from both lasso regressions for our post-lasso OLS regression. An intuition for this approach is that confounders are variables that affect both `d` and `y`. To ensure that very few confounders are omitted, it thus seems not implausible to use as a broad control set all variables that relevantly affect `d` or `y`.

Let us apply the post double selection method:

```
# post double selection
# 1. run the two lasso regressions
d = mat[,2]
lasso.dx = gamlr(y=d,x=X[,-2])
lasso.yx = gamlr(y=y,x=X[,-2])

# 2. compute union of selected variables from
# both lasso regressions
vars1 = names(lasso_coef(lasso.dx))
vars2 = names(lasso_coef(lasso.yx))
vars = union(vars1,vars2)

# 3. Run OLS estimation with d and all selected variables
post.lasso = lm.fit(y=y,x=X[,c("const","d",vars)])
coef(post.lasso)[2] # looks fairly unbiased
```

```
##         d
## 0.9643243
```

Now our estimator looks fairly unbiased. Let us look at the selected variables:

```
vars_counts(vars1)
```

```
## num.vars       xc       xe       xu       xy
##      148       50        1       96        1
```

```
vars_counts(vars2)
```

```
## num.vars       xc       xe       xu       xy
##      207       50        1      106       50
```

```
vars_counts(vars)
```

```
## num.vars       xc       xe       xu       xy
##      271       50        1      170       50
```

We see that the post double selection method selects all 50 confounders `xc`. Interestingly the confounders are found in both the first and second lasso regression. This means if we don't add the variable of interest `d` when regressing `y` on all potential controls, we seem more likely to pick up confounders.

You may think: “But we selected many more variables: 271 instead of only 60 before. In particular, we wrongly picked up 170 unrelated variables. Therefore it is no surprise that we now also select the 50 confounders.”

However, Belloni et. al. (2014) actually use a different method to select the penalty parameter $\lambda$, not the AICc critierion used in gamlr. The method is implemented in the R package hdm authored by Martin Spindler, Victor Chernozhukov and Chris Hansen. Let us repeat the post double selection procedure using the function `rlasso` from the `hdm` package:

```
library(hdm)
# post double selection
# 1. run the two lasso regressions
lasso.dx = rlasso(y=d,x=X[,-2])
lasso.yx = rlasso(y=y,x=X[,-2])

# 2. compute union of selected variables from
# both lasso regressions
vars1 = names(lasso_coef(lasso.dx))
vars2 = names(lasso_coef(lasso.yx))
vars = union(vars1,vars2)

# 3. Run OLS estimation with d and all selected variables
post.lasso = lm.fit(y=y,x=X[,c("const","d",vars)])
coef(post.lasso)[2] # looks unbiased
```

```
##        d
## 1.013494
```

```
# Var counts
vars_counts(vars)
```

```
## num.vars       xc       xe       xu       xy
##      113       50        1       12       50
```

We now only select 12 unrelated variables but still all confounders. Correspondingly, our resulting OLS estimate is pretty close to the true causal effect.

The function rlassoEffect is a convenient wrapper to directly apply the double selection method via `rlasso`:

```
rlassoEffect(x=X[,-2],d=d,y=y,method="double selection")
```

```
##
## Call:
## rlassoEffect(x = X[, -2], y = y, d = d, method = "double selection")
##
## Coefficients:
##    d1
## 1.013
```

The `hdm` package also implements an alternative “partialling out” method based on an idea related to regression anatomy and the FWL theorem. The method is well explained in Chapter 4 of the hdm vignette. Here we find:

```
rlassoEffect(x=X[,-2],d=d,y=y,method="partialling out")
```

```
##
## Call:
## rlassoEffect(x = X[, -2], y = y, d = d, method = "partialling out")
##
## Coefficients:
## [1]  0.9402
```

While the estimated $\hat \alpha$ differs slightly, in all simulations where I checked either both approaches seemed to worked well or both failed.

Cases were post double selection fails but selection with single lasso regression works well

While in the previous example post double selection worked very well, I believe there are cases in which selection with a single lasso regression on `y` works better for selecting variables.

We now simulate data from our model with the main difference that we now also add $K_e=50$ variables that are sources of exogenous variation, i.e. they only affect $d$ but not $y$.

```
set.seed(1)
mat = lasso_sim(alpha=1,n=700,Kc=50,Ke=50,Ky=50,Ku=700,return.what = "data")
y = mat[,1]
d = mat[,2]
X = cbind(1,mat[,-1]); colnames(X)[1] = "const"

# Short OLS regression y on d: biased
coef(lm.fit(y=y,x=X[,1:2]))[2]
```

```
##        d
## 1.472816
```

Now let’s compare two linear regressions where in both we select 49 of the 50 confounders. In one regression we also control for all 50 observable sources of exogenous variation `xe` in the other we don’t add any `xe` as control variable. Make a guess about the biases of $\hat \alpha$ in the two regressions before you look at the results here:

```
xc.cols = 3:52; xe.cols = 53:102
# Controlling for 49 of 50 of confounders,
# not controlling sources of exogenous variation
coef(lm.fit(y=y,x=X[,c(1:2, xc.cols[1:49])]))[2]
```

```
##        d
## 1.028753
```

```
# Controlling for 49 of 50 of confounders
# and controlling for all sources of exogenous variation
coef(lm.fit(y=y,x=X[,c(1:2, xc.cols[1:49], xe.cols)]))[2]
```

```
##        d
## 1.478361
```

If we don’t add any observable source of exogenous variation, we find an estimate pretty close to the true causal effect $\alpha=1$, but if we control for the exogenous variation our estimate looks substantially biased.

OK, one probably should not infer a bias from a single simulation run. Therefore, I have run a proper Monte-Carlo simulation, where I repeated the data generation and both estimations a 1000 times. The code and results can be found on Github. Let’s take a look at the distribution of the simulated estimators $\hat \alpha$ in both cases:

```
sim = readRDS("control_exo_sim.Rds")
library(ggplot2)
ggplot(sim, aes(x=alpha.hat, fill=reg)) + geom_density() +
  facet_wrap(~reg) + geom_vline(xintercept=1, alpha=0.7)
```

```
sim %>%
  group_by(reg) %>%
  summarize(
    bias = mean(alpha.hat-alpha),
    se = sd(alpha.hat)
  )
```

```
## # A tibble: 2 x 3
##   reg               bias     se
##
## 1 control_xe      0.507  0.205
## 2 dont_control_xe 0.0250 0.0437
```

We now properly see that we have a substantial bias in our causal effect estimate $\hat \alpha$ if we control for 49 out of 50 confounders and also add all sources of exogenous variation `xe` as control variables. The standard error is also high. If we don't control for the sources of exogenous variation both the bias and standard error of $\hat \alpha$ are very small instead.

The following graph thus summarizes the goal in selecting control variables:

Unless we can manually guarantee that enough relevant sources of exogenous variation are excluded from the pool of candidate control variables, there seems to be a drawback of the post-double selection procedure: The lasso regression of $d$ on all potential control variables is likely to select the sources of exogenous variation, which we don't want to use as control variables.

In contrast, if we would just select variables with a single lasso regression of $y$ on $d$ and all potential control variables, we seem much less likely to select sources of exogenous variation, since by definition they only affect $y$ via $d$. Let's run both approaches on our simulated data set:

```
# Post double selection
double_sel = rlassoEffect(x=X[,-2],y=y,d=d, method="double selection")
coef(double_sel)
```

```
##       d1
## 1.440466
```

```
# Simple gamlr post-lasso estimator
simple_lasso = gamlr(x=X,y=y)
post_lasso_coef(simple_lasso,x=X,y=y,keep.intercept = FALSE)[1]
```

```
##        d
## 1.023805
```

Indeed, the simple lasso approach yields here a much better estimate of the causal effect than double selection. The `hdm` package also allows to compute confidence intervals:

```
confint(double_sel,level = 0.999)
```

```
##      0.05 %  99.95 %
## d1 1.334566 1.546366
```

The computed 99.9% confidence interval of the post double selection estimator is erroneously far away from the true causal effect $\alpha=1$.

The following code estimates both models again and also returns statistics about the selected variables:

```
set.seed(1)
lasso_sim(alpha=1,n=700,Kc=50,Ke=50,Ky=50,Ku=700,
  models=c("gamlr_simple","rlasso_double_sel"))
```

```
##               model     coef num.vars xc xe xu xy
## 1      gamlr_simple 1.023805      179 50 28 51 50
## 2 rlasso_double_sel 1.440466        7  5  2  0  0
```

We see here that the reason why the post double selection algorithm does not work in our example is not that too many `xe` are selected, but that overall only 7 variables are selected including just 5 of the 50 confounders. The default settings of `rlasso` seem to select relatively few variables.

Below we estimate two variants of the post double selection where we reduce a parameter `c` in the determination of the penalty term. This causes `rlasso` to select more variables:

```
set.seed(1)
models = list(
  rlasso_double_sel_c106 = list(lasso.fun="rlasso",type="double_sel",
    args=list(penalty=list(c=1.06))),
  rlasso_double_sel_c100 = list(lasso.fun="rlasso",type="double_sel",
    args=list(penalty=list(c=1)))
)
lasso_sim(alpha=1,n=700,Kc=50,Ke=50,Ky=50,Ku=700,return.what = "details",
  models = models)
```

```
##                    model     coef num.vars xc xe xu xy
## 1 rlasso_double_sel_c106 1.489470       76 32 34  8  2
## 2 rlasso_double_sel_c100 1.214481      140 50 50 34  6
```

Now more variables are selected, but never more confounders `xc` than sources or exogenous variation `xe`. Let's load the results of a proper Monte-Carlo simulation with 1000 simulation runs:

```
sim = readRDS("lasso_sel_sim.Rds")
sim %>%
  group_by(model) %>%
  summarize(
    bias = mean(coef-1),
    se = sd(coef)
  )
```

```
## # A tibble: 4 x 3
##   model                    bias      se
##
## 1 gamlr_simple           0.0242 0.00951
## 2 rlasso_double_sel      0.431  0.101
## 3 rlasso_double_sel_c100 0.176  0.292
## 4 rlasso_double_sel_c106 0.365  0.192
```

We see how the simple lasso model yields the lowest bias and also has a lower standard error than the 3 specifications of the post double selection approach.

Are confounders that strongly affect y worse than those that strongly affect d?

Finally, let us look at a simulation where both the `xe` and the `xc` variables are confounders. Yet, there is a difference. The confounders `xe` strongly affect the explanatory variable `d` with coefficients $\beta_k^{ed}=10$ and only weakly affect `y` with $\beta_k^{ey}=0.5$. The coefficients shall be reversed for the confounders `xc` that strongly affect `y` with $\beta_k^{cy}=10$ but only weakly affect `d` with $\beta_k^{cd}=0.5$.

Let's compare post double selection with a simple `gamlr` lasso selection in this setting:

```
set.seed(1)
```

```
res = lasso_sim(alpha=1,n=100,Kc=10,Ke=10,Ky=5,Ku=20, beta.ed = 10,
  beta.ey = 0.5, beta.cd = 0.5, beta.cy = 10,
  models=c("gamlr_simple","rlasso_double_sel"))
select(res, - coef)
```

```
##                 model num.vars xc xe xu xy
## 1      gamlr_simple       16 10  1  0  5
## 2 rlasso_double_sel       17  7 10  0  0
```

I have not yet shown the estimated coefficients. But we see that the simple lasso regression has selected all confounders that strongly affect `y` but only 1 confounder that strongly affects `d`. In contrast, double selection picks 7 out of the 10 confounders that strongly affect `y` and all 10 that strongly affect `d`.

Why not make a guess about the two estimates $\hat \alpha$ of the two methods…

Loading…

Ok, let's look at the complete result:

```
res
```

```
##                 model     coef num.vars xc xe xu xy
## 1      gamlr_simple 1.051267       16 10  1  0  5
## 2 rlasso_double_sel 8.150202       17  7 10  0  0
```

While the simple lasso estimate is pretty close to the true causal effect, the double selection estimate is pretty far away, even though it selects many more confounders. It seems as if omitting confounders that strongly affect the dependent variable is worse than omitting those that strongly affect the explanatory variable.

If you find this result surprising, it may be because the simple omitted variable bias formula is often presented in a way that could wrongly suggest that an omitted confounder that strongly affects the explanatory variable `d` is equally bad as one that strongly affects the dependent variable `y`. However, even with a single omitted variable it is worse if it strongly affects `y` rather than `d` (see my previous blog post for more details).

A thousand simulation runs also verify that the single run above is representative:

```
sim = readRDS("lasso_sim3.Rds")
sim %>%
  group_by(model) %>%
  summarize(
    bias = mean(coef-1),
    se = sd(coef),
    num.vars = mean(num.vars),
    xe = mean(xe),
    xc = mean(xc)
  )
```

```
## # A tibble: 2 x 6
##   model               bias      se num.vars     xe    xc
##
## 1 gamlr_simple      0.0489 0.00393     16.6  0.685 10
## 2 rlasso_double_sel 6.85   4.04        17.8 10      6.65
```

To sum up, this simulation provides another related example, in which control variable selection with a simple lasso regression works better than the post double selection method.

Conclusion

We first have illustrated a setting in which post double selection performed considerably better in selecting control variables than just a simple single lasso regression. Post double selection is probably most save to apply in settings where an exogenous source of variation is known and manually taken care that it is excluded from the set of potential control variables. Intuitively, the procedure seems also reasonable in quasi-experimental settings where `d` is a dummy variable of a single policy change. Belloni et. al. (2014) indeed mostly refer to such settings.

Yet, we also have illustrated that in situations in which many important sources of exogenous variation are part of the set of potential control variables, a simple lasso regression can outperform post double selection. We also exemplified in a setting without observable sources of exogenous variation that a simple lasso regression was better able to select the worse confounders (i.e. those that affect `y` strongly).

Perhaps a method will surface that works well across all situations were either double selection or selection with a simple lasso fails. Alas, I imagine that the selection of control variables can be automatized only up to a certain extend and will continue to require considerable subjective assessments. For example, in addition to the issues discussed in this post, one must also ensure that the set of potential control variables does not contain colliders or channel variables (unless one wants to explicitly ignore the mediated effect from certain channels).