Replication Beg. et al. (2022)

Pedro Scatimburgo

2023-02-27

Preamble

I will replicate Table 1 - Summary Statistics in page 75 and columns (1), (3) and (5) from Panel A. Table 2 - Achievement Effects in page 76. Unfortunately, I wasn't able to replicate any of the LASSO results in Panel B. Still, I present the code below and compare with the selected controls to offer possible explanations.

```
# Loads main packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.1     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.4     v forcats 1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(estimatr)
```

Table 1 - Summary statistics

```
# Loads data
elearn_balance_data <- haven::read_dta("data/Elearn/elearn_balance_data.dta")

# Columns 1 and 2
table1_panelA_columns12 <- elearn_balance_data %>%
  group_by(treatment) %>%
  summarise(
    across(
      c(
        z_score_total_bl,
        age_bl,
        attendance_bl,
        computer_yn_bl,
        m_ed_noschool,
        f_ed_noschool
      ),
      list(mean = ~ mean(.x, na.rm=T),
           sd = ~ sd(.x, na.rm=T))
    )
  ) %>%
```

```r
  pivot_longer(-treatment,
               names_to = "var",
               values_to = "value") %>%
  pivot_wider(names_from = treatment, values_from = value) %>%
  rename(
    variable = var,
    control = `0`,
    treatment = `1`
  )

# Column 3
table1_panelA_column3_models <- elearn_balance_data %>%
  filter(child_interviewed_bl==1) %>%
  select(
    school_code,
    treatment,
    z_score_total_bl,
    age_bl,
    attendance_bl,
    computer_yn_bl,
    m_ed_noschool,
    f_ed_noschool
  ) %>%
  pivot_longer(c(-school_code, -treatment),
               names_to = "variable",
               values_to = "value") %>%
  group_by(variable) %>%
  nest() %>%
  mutate(
    model = purrr::map(data,
                       ~ estimatr::lm_robust(
                         data = .,
                         formula = value ~ treatment,
                         cluster = school_code,
                         se_type = "stata"))
  )

table1_panelA_column3 <- cbind(table1_panelA_column3_models$variable,
                               purrr::map_dfr(
                                 .x =table1_panelA_column3_models$model,
                                 .f = broom::tidy) %>% filter(term=="treatment")
                               ) %>%
  select(`table1_panelA_column3_models$variable`,
         estimate,
         `std.error`,
         `p.value`)

(knitr::kable(table1_panelA_columns12,
              caption = "Table 1 Panel A Columns 1 and 2"))
```

Table 1: Table 1 Panel A Columns 1 and 2

variable	control	treatment
z_score_total_bl_mean	0.0679137	-0.0559342
z_score_total_bl_sd	1.0100126	0.9860643
age_bl_mean	13.8696296	13.8962766
age_bl_sd	1.2305022	1.2374050
attendance_bl_mean	1.1674074	1.4946809
attendance_bl_sd	1.7877273	2.4094738
computer_yn_bl_mean	0.4074074	0.4268617
computer_yn_bl_sd	0.4917162	0.4949511
m_ed_noschool_mean	0.3333333	0.3523936
m_ed_noschool_sd	0.4717541	0.4780337
f_ed_noschool_mean	0.1970370	0.1728723
f_ed_noschool_sd	0.3980555	0.3783885

```
(knitr::kable(table1_panelA_column3,
              caption = "Table 1 Panel A Column 3"))
```

Table 2: Table 1 Panel A Column 3

table1_panelA_column3_models$variable	estimate	std.error	p.value
z_score_total_bl	-0.1240815	0.1904352	0.5172533
age_bl	0.0266470	0.1018226	0.7944799
attendance_bl	0.3272734	0.1732048	0.0638243
computer_yn_bl	0.0194543	0.0484134	0.6892807
m_ed_noschool	0.0190603	0.0576065	0.7419346
f_ed_noschool	-0.0241647	0.0340920	0.4812821

Table 2 - Achievement Scores

```
# Loads data
elearn_reg_data <- haven::read_dta("data/Elearn/elearn_reg_data.dta") %>%
  rename_with(.cols = starts_with("_"), ~ stringr::str_replace(.x, "_", "v_"))
```

Remark: there is no available dataset to replicate the results of column (2).

```
# ---- Panel A Col 1-4 ----

# Clasrooms
project_classrooms <- elearn_reg_data %>%
  filter(tooktest_el==1) %>%
  estimatr::lm_robust(
  formula = z_irt_total_el ~
    treatment +
    v_z_irt_math_bl +
    v_z_irt_sci_bl +
    strataFE1 +
    strataFE2 +
    strataFE3 +
    strataFE4 +
    strataFE5,
```

```
  clusters = school_code,
  se_type = "stata"
) %>%
  broom::tidy() %>%
  select(1,2,4)

project_clasrooms_group_mean <- elearn_reg_data %>%
  filter(treatment==0 & tooktest_el==1) %>%
  summarise(mean = mean(z_irt_total_el, na.rm=T))

# Column 2
pec_classrooms <- elearn_reg_data %>%
  filter(tooktest_el==1 & took_std==1) %>%
  estimatr::lm_robust(
    formula = z_scoreindex_el ~
      treatment +
      v_z_irt_math_bl +
      v_z_irt_sci_bl +
      v_meanmath_pec_2016 +
      v_meansci_pec_2016 +
      v_meaneng_pec_2016 +
      v_meaneng_pec_2016_mi +
      strataFE1 +
      strataFE2 +
      strataFE3 +
      strataFE4 +
      strataFE5,
    clusters = school_code,
    se_type = "stata"
  )%>%
  broom::tidy() %>%
  select(1,2,4)

pec_clasrooms_group_mean <- elearn_reg_data %>%
  filter(treatment==0 & tooktest_el==1 & took_std==1) %>%
  summarise(mean = mean(z_scoreindex_el, na.rm=T))

(knitr::kable(project_classrooms,
              caption = "Table 2 Panel A Column 1"))
```

Table 3: Table 2 Panel A Column 1

term	estimate	statistic
(Intercept)	0.9856713	4.393863
treatment	0.2556772	1.896638
v_z_irt_math_bl	0.2703520	5.889213
v_z_irt_sci_bl	0.1609753	4.140037
strataFE1	-0.5963266	-1.784091
strataFE2	-0.8529614	-3.071121
strataFE3	-0.6507201	-2.557387
strataFE4	-0.4738788	-1.939964
strataFE5	-0.6764262	-2.526221

```
(knitr::kable(project_clasrooms_group_mean,
             caption = "Project Average control group change or mean"))
```

Table 4: Project Average control group change or mean

mean
0.4917922

```
(knitr::kable(pec_classrooms,
             caption = "Table 2 Panel A Column 3"))
```

Table 5: Table 2 Panel A Column 3

term	estimate	statistic
(Intercept)	0.1054971	0.5061549
treatment	0.2690645	2.2653021
v__z__irt__math__bl	0.2153490	6.4781188
v__z__irt__sci__bl	0.1196785	3.2452808
v__meanmath__pec__2016	0.1056084	1.2676751
v__meansci__pec__2016	0.0139204	0.1387179
v__meaneng__pec__2016	-0.0618132	-0.7482025
v__meaneng__pec__2016__mi	0.0267513	0.3291950
strataFE1	-0.5893693	-2.2509843
strataFE2	-0.5045070	-2.0457491
strataFE3	-0.0612791	-0.2127759
strataFE4	0.3168674	1.4224077
strataFE5	-0.4533764	-1.6182046

```
(knitr::kable(pec_clasrooms_group_mean,
             caption = "Combined project and PEC Average control group change or mean"))
```

Table 6: Combined project and PEC Average control group change or mean

mean
0

LASSO

Here I show my attempts to replicate the results in Panel B, which includes additional controls selected by the post-double LASSO. In the Online Appendix, the authors mention that set of potential controls have 298 variables. For eLearn Classrooms, LASSO selected: teacher employment rank, time spent on non-classroom duties and extra classes, mothers occupations, and parents rela- tionship status. Below, I show the Stata code made available by the authors that is responsible for estimating the coefficient in Panel B Column (1):

```
foreach outcome of varlist z_irt_total_el {
pdslasso `outcome' treatment (`prepped' $strata ) if `conditions_proj',
partial($strata $partialled_proj) cluster(school_code)
outreg2 using PanelB, replace dta label keep(treatment*)
}
```

z_irt_total_el is outcome of interest. The second line regresses this outcome against the treatment variable using the pdslasso command. According to the pdslasso documentation, (`prepped' $strata ) the set of potential controls. They are already standardized. if `conditions_proj simply tells Stata to select only students that took the test, that is, tooktest_el==1. partial($strata $partialled_proj) indicates which variables are always to be included in the model: all strata dummys and the student's scores in math and science, _z_irt_math_bl and _z_irt_sci_bl, respectively. Finally, they cluster at the school_code.

I attempted to replicate the result using two packages: glmnet and hdm.

Using glmnet

First I filter the dataframe to inlclude only the students who took the test:

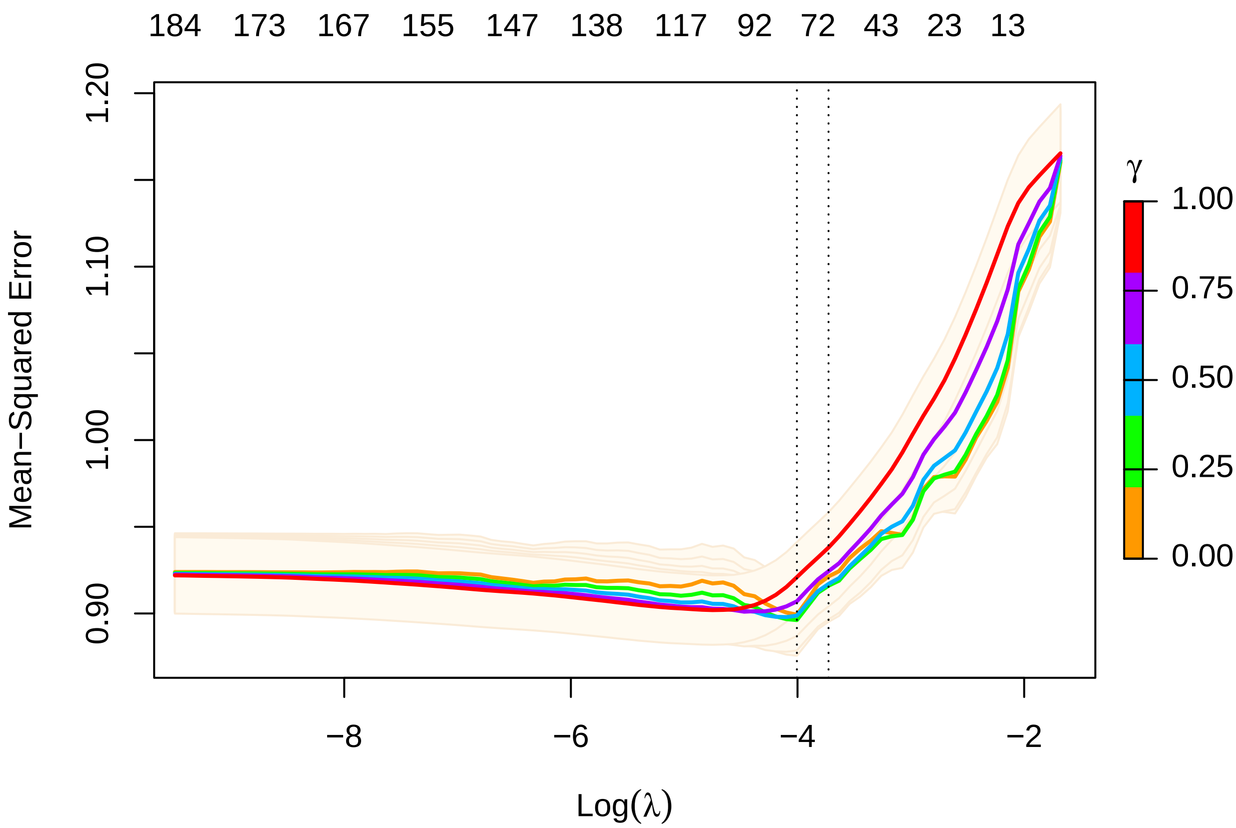
```
elearn_reg_data_tooktest <- elearn_reg_data %>%
  filter(tooktest_el==1)
```

Then I create the matrix including all potential controls. I remove strataFE6to avoid perfect collinearity, like I did before. I also select the variables in a specific order that allows me to easily handle the penalty.factor argument in the next chunk of code. Notice that elearn_reg_data_lasso_matrix has 305 variables, of which 7 are always to be included in the model and the remaining 298 is the number of items in the set of potential controls. So the set of potential controls *is not* the reason why the replication fails.

```
elearn_reg_data_lasso_matrix <- elearn_reg_data_tooktest %>%
  select(v_z_irt_math_bl,
         v_z_irt_sci_bl,
         starts_with("strata"),
         starts_with("v_"),
         -strataFE6) %>%
  as.matrix()
```

Then I perform the cross-validation using glmnet::cv.glmnet. x is the set of potential controls, y is the outcome of interest, alpha = 1 represents a LASSO regularization, relax=Tallows me to select gamma parameters in the regularization, and the penalty.factor indicates which variables are to be always included in the model.

```
plot(cv_model)
```



The value of λ that gives minimum mean cross-validated error is `0.0086541`. Then I select optimal controls using `glmnet::glmnet` for this value of λ:

```
best_model <- glmnet::glmnet(
  x = elearn_reg_data_lasso_matrix,
  y = elearn_reg_data_tooktest %>% pull(z_irt_total_el),
  lambda = cv_model$lambda.min,
  alpha = 1,
  penalty.factor = c(rep(0, 7), rep(1, 298)),
  relax = T
)

selected_coef <- as.data.frame(as.matrix(best_model$beta)) %>%
  filter(s0 != 0)
```

This method selects a total of `101` coefficients, which is a much larger set than the one found by the authors. I am not sure why this is happening. `glmnet::cv.glmnet` should have given us a much larger λ that minimizes the MSE so that we would have a number of controls similar to the one found by the authors. I ran `best_model` for every λ in the sequence and none of them resulted in the exact same set of controls found by the authors.

Using `hdm`

(Chernozhukov, Hansen, and Spindler 2016) published the `hdm` package on CRAN which performs post-double selection LASSO. It is much more similar to the usage of `pdslasso`in Stata, in comparison to `glmnet`. It also comes with a very neat documentation by the authors themselves that can be read here. Following their instructions, I use the `hdm::rlassoEffect` function. `x`is the set of potential controls, `y` is the outcome of interest, and `d` is the main regressor that is treated as causal. I use `I3 = c(rep(T, 8), rep(F, 298))` so the strata dummies and the two main controls are always included in the model, like I did when using `glmnet`. I explictly asks for the double selection method using `method = "double selection"`.

```
lasso_x <- elearn_reg_data_tooktest %>%
  select(v_z_irt_math_bl,
         v_z_irt_sci_bl,
         starts_with("strata"),
         starts_with("v_"),
         -strataFE6) %>%
  as.matrix()

lasso_y <- elearn_reg_data_tooktest %>% pull(z_irt_total_el) %>% as.matrix()

lasso_d <- elearn_reg_data_tooktest %>% pull(treatment) %>% as.matrix()

lasso <- hdm::rlassoEffect(
  x = lasso_x,
  y = lasso_y,
  d = lasso_d,
  I3 = c(rep(T, 7), rep(F, 298)),
  method = "double selection"
)
```

However:

```
summary(lasso)
```

```
## [1] "Estimates and significance testing of the effect of target variables"
##    Estimate. Std. Error t value Pr(>|t|)
## d1 6.037e-02  2.283e+11       0        1
```

```
(lasso$no.selected)
```

```
## [1] 0
```

My guess is that I was including all levels of some dummy variable, resulting in perfect collinearity that explain the standard error. However, the set of potential controls that I am using is exactly the same used by the authors. Besides, since I have no access to the survey, I cannot say which columns represent different levels of the same dummy variable based solely on their names.

Another possibility is that the 298 potential controls are receiving some treatment before the authors run the `pdslasso` command. Some kind of treatment is surely being conducted, since I can see the command in the `master.do` file. If this treatment somehow makes the `hdm::rlassoEffect` behaves erratically, this could explain the results. Unfortunately, I have no access to the variables before this treament, neither I can know for certainty what exactly this treatment is doing.

Finally, I can at least be sure that the problem is in the set of potential controls, since `hdm::rlassoEffect` gives a reasonable result if I exclude them:

```
lasso_x <- elearn_reg_data_tooktest %>%
  select(v_z_irt_math_bl,
         v_z_irt_sci_bl,
         starts_with("strata"),
         # starts_with("v_"),
         -strataFE6) %>%
  as.matrix()
```

```r
lasso_y <- elearn_reg_data_tooktest %>% pull(z_irt_total_el) %>% as.matrix()

lasso_d <- elearn_reg_data_tooktest %>% pull(treatment) %>% as.matrix()

lasso <- hdm::rlassoEffect(
  x = lasso_x,
  y = lasso_y,
  d = lasso_d,
  # I3 = c(rep(T, 7), rep(F, 298)),
  method = "double selection"
)

summary(lasso)
```

```
## [1] "Estimates and significance testing of the effect of target variables"
##    Estimate. Std. Error t value Pr(>|t|)
## d1   0.24551    0.04194   5.854 4.81e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And selecting only the students' scores and their square values:

```r
lasso_x <- elearn_reg_data_tooktest %>%
  select(v_z_irt_math_bl,
         v_z_irt_sci_bl,
         starts_with("strata"),
         starts_with("v_z_"),
         -strataFE6) %>%
  as.matrix()

lasso_y <- elearn_reg_data_tooktest %>% pull(z_irt_total_el) %>% as.matrix()

lasso_d <- elearn_reg_data_tooktest %>% pull(treatment) %>% as.matrix()

lasso <- hdm::rlassoEffect(
  x = lasso_x,
  y = lasso_y,
  d = lasso_d,
  I3 = c(rep(T, 5), rep(F, 6)),
  method = "double selection"
)

summary(lasso)
```

```
## [1] "Estimates and significance testing of the effect of target variables"
##    Estimate. Std. Error t value Pr(>|t|)
## d1   0.24307    0.04156   5.849 4.96e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
(lasso$no.selected)
```

```
## [1] 0
```

Again a reasonable result. Thus I am convinced that the problem resides with the dummy variables. However, in this last specification, `hdm::rlassoEffect` again selects no controls.

Chernozhukov, Victor, Chris Hansen, and Martin Spindler. 2016. "High-Dimensional Metrics in r." arXiv. https://doi.org/10.48550/ARXIV.1603.01700.