Apéndice. Codigo PS2 (STATA)

2023-08-17

```
/*******************************************************************************
                          Semana 3: Problem Set 2

                          Universidad de San Andrés
                              Economía Aplicada
                                   2022

                        Barnes, Fasan, Legaspe y Martin
*******************************************************************************/


* Source: https://www.aeaweb.org/articles?id=10.1257/app.20200204


/*******************************************************************************
Este archivo sigue la siguiente estructura:

0) Set up environment and globals

1) Regressions

2) Correcciones de pvalues y/o intervalos de confianza

*******************************************************************************/



* 0) Set up environment
*==============================================================================*
clear all
gl main "C:/Users/Usuario/Desktop/MAESTRIA/Economia Aplicada/TPs/PS2"
gl input "$main/input"
gl output "$main/output"

* Open data set

use "$input/measures.dta", clear

* Global with control variables

global covs_eva "male i.eva_fu"
global covs_ent "male i.ent_fu"

*Recode treatment variable
```

```
replace treat = 0 if treat == 1
replace treat = 1 if treat == 2


* 1) Table 2 Replication
*==============================================================================*

******************************************************************************
* PANEL A (Child's cognitive skills at follow up)
******************************************************************************
* 1) Replicamos la tabla 2
local bayley "b_tot_cog b_tot_lr b_tot_le b_tot_mf"
foreach y of local bayley{
local append append
if "`y'"=="b_tot_cog" local append replace
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_eva , cluster(cod_dane)
}

local macarthur "mac_words mac_phrases"
foreach y of local macarthur{
    cap drop V*
    reg `y'1_st treat mac_words0_st $covs_ent , cluster(cod_dane)
}



******************************************************************************
* PANEL B (Child's socio-emotional skills at follow up)
******************************************************************************

local bates "bates_difficult bates_unsociable bates_unstoppable"
foreach y of local bates{
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_ent, cl(cod_dane)
}

local roth "roth_inhibit roth_attention"
foreach y of local roth{
    cap drop V*
    reg `y'1_st treat bates_difficult0_st $covs_ent , cluster(cod_dane)
}


******************************************************************************
* PANEL C (Material investments)
******************************************************************************

local fcimat "fci_play_mat_type Npaintbooks Nthingsmove Ntoysshape Ntoysbought"
foreach y of local fcimat{
    cap drop V*
    reg `y'1_st treat fci_play_mat_type0_st $covs_ent , cluster(cod_dane)
}
```

```
*******************************************************************************
* PANEL D (Time investments)
*******************************************************************************
local fcitime "fci_play_act home_stories home_read home_toys home_name"
foreach y of local fcitime{
    cap drop V*
    reg `y'1_st treat fci_play_act0_st $covs_ent , cluster(cod_dane)
}



* 2) Correcion pur multiples hipotesis

* 2) Multiple hypotesis correction
*==============================================================================*
* 2.1) Bonferroni Correction
*==============================================================================*
* Remove index from list of variables
global bayley "b_tot_cog b_tot_lr b_tot_le b_tot_mf"
global macarthur "mac_words mac_phrases"
global bates "bates_difficult bates_unsociable bates_unstoppable"
global roth "roth_inhibit roth_attention"
global fcimat "fci_play_mat_type Npaintbooks Nthingsmove Ntoysshape Ntoysbought"
global fcitime "fci_play_act home_stories home_read home_toys home_name"

* Store in a scalar the number of hypothesis tested
scalar hyp = 21

* Run regressions storing p-value
eststo clear
* Panel A
foreach y in $bayley{
    local append append
    if "`y'"=="b_tot_cog" local append replace
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_eva , cluster(cod_dane)
    eststo: test treat = 0
    estadd scalar p_value = r(p)
    estadd scalar corr_p_value = min(1,r(p)*hyp) // Add corrected p-value

}

foreach y in $macarthur{
    cap drop V*
    reg `y'1_st treat mac_words0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    estadd scalar p_value = r(p)
    estadd scalar corr_p_value = min(1,r(p)*hyp) // Add corrected p-value

}
* Panel B
foreach y in $bates{
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_ent, cl(cod_dane)
    eststo: test treat = 0
    estadd scalar p_value = r(p)
    estadd scalar corr_p_value = min(1,r(p)*hyp) // Add corrected p-value
}

foreach y in $roth{
    cap drop V*
    reg `y'1_st treat bates_difficult0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    estadd scalar p_value = r(p)
    estadd scalar corr_p_value = min(1,r(p)*hyp) // Add corrected p-value
}

* Panel C
foreach y in $fcimat{
    cap drop V*
    reg `y'1_st treat fci_play_mat_type0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    estadd scalar p_value = r(p)
    estadd scalar corr_p_value = min(1,r(p)*hyp) // Add corrected p-value
}

* Panel D
foreach y in $fcitime{
    cap drop V*
    reg `y'1_st treat fci_play_act0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    estadd scalar p_value = r(p)
    estadd scalar corr_p_value = min(1,r(p)*hyp) // Add corrected p-value
}


esttab using "$output/Table 2_bonferroni.txt", p se replace label noobs ///
keep(treat, relax) ///
cells(b(fmt(2) star) se(par fmt(2))) ///
stats(p_value corr_p_value blank N r2, fmt(2 2 0 2) labels("P-value" "Corrected
                                                           p-value" " "  "Number
                                                           of Observations"
                                                           "R-Squared"))

*==============================================================================*
* 2.2) Holm Correction
*==============================================================================*

* Define number of hypothesis
scalar hyp = 21
* Define level of significance
scalar signif = 0.05

* Store p-values in matrix
mat p_values = J(21,1,.)
```

```
* Panel A
scalar i = 1
 quietly foreach y in $bayley{
    local append append
    if "`y'"=="b_tot_cog" local append replace
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_eva , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1

}

scalar i = 5
quietly foreach y in $macarthur{
    cap drop V*
    reg `y'1_st treat mac_words0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}
* Panel B
scalar i = 7
 quietly foreach y in $bates{
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_ent, cl(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}
scalar i = 10
quietly foreach y in $roth{
    cap drop V*
    reg `y'1_st treat bates_difficult0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}

* Panel C
scalar i = 12
quietly foreach y in $fcimat{
    cap drop V*
    reg `y'1_st treat fci_play_mat_type0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}
* Panel D
scalar i = 17
quietly foreach y in $fcitime{
    cap drop V*
    reg `y'1_st treat fci_play_act0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}


preserve

clear

svmat p_values


gen var = _n
sort p_values1

gen alpha_corr = signif/(hyp+1-_n)

gen significant = (p_values1<alpha_corr)

replace significant = 0 if significant[_n-1]==0

sort var

export delimited using "$main/output/Holm_alphas.csv", replace
// guardamos como csv los coef ajustados

restore



*================================================================================*
* 2.3) Benjamini, Krieger, and Yekutieli (2006)s' correction
*================================================================================*


* First, run regressions and keep p-values
preserve // pasarlo en la consola
scalar i = 1
mat p_values = J(21,1,.)
* Panel A
scalar i = 1
 quietly foreach y in $bayley{
    local append append
    if "`y'"=="b_tot_cog" local append replace
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_eva , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1

}
```

```
scalar i = 5
quietly foreach y in $macarthur{
    cap drop V*
    reg `y'1_st treat mac_words0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}
* Panel B
scalar i = 7
 quietly foreach y in $bates{
    cap drop V*
    reg `y'1_st treat `y'0_st $covs_ent, cl(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}
scalar i = 10
quietly foreach y in $roth{
    cap drop V*
    reg `y'1_st treat bates_difficult0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}

* Panel C
scalar i = 12
quietly foreach y in $fcimat{
    cap drop V*
    reg `y'1_st treat fci_play_mat_type0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}
* Panel D
scalar i = 17
quietly foreach y in $fcitime{
    cap drop V*
    reg `y'1_st treat fci_play_act0_st $covs_ent , cluster(cod_dane)
    eststo: test treat = 0
    mat p_values[i,1]=r(p)
scalar i = i + 1
}


*****
clear
svmat p_values
gen outcome = _n
rename p_values1 pval
save "$output/pvals.dta", replace
restore
```

```
**** Now use Michael Anderson's code for sharpened q-values

use "$output/pvals.dta", clear
version 10
set more off

* Collect the total number of p-values tested
quietly sum pval
local totalpvals = r(N)

* Sort the p-values in ascending order and generate a variable that codes
*       each p-value's rank
quietly gen int original_sorting_order = _n
quietly sort pval
quietly gen int rank = _n if pval~=.

* Set the initial counter to 1
local qval = 1

* Generate the variable that will contain the BKY (2006) sharpened q-values
gen bky06_qval = 1 if pval~=.

* Set up a loop that begins by checking which hypotheses are rejected at
q = 1.000, then checks which hypotheses are rejected at q = 0.999, then checks
which hypotheses are rejected at q = 0.998, etc.  The loop ends by checking
which hypotheses are rejected at q = 0.001.

while `qval' > 0 {
    * First Stage
    * Generate the adjusted first stage q level we are testing: q' = q/1+q
    local qval_adj = `qval'/(1+`qval')
    * Generate value q'*r/M
    gen fdr_temp1 = `qval_adj'*rank/`totalpvals'
    * Generate binary variable checking condition p(r) <= q'*r/M
    gen reject_temp1 = (fdr_temp1>=pval) if pval~=.
    * Generate variable containing p-value ranks for all p-values that meet above
      *condition
    gen reject_rank1 = reject_temp1*rank
    * Record the rank of the largest p-value that meets above condition
    egen total_rejected1 = max(reject_rank1)

    * Second Stage
    * Generate the second stage q level that accounts for hypotheses rejected in
    first stage: q_2st = q'*(M/m0)
    local qval_2st = `qval_adj'*(`totalpvals'/(`totalpvals'-total_rejected1[1]))
    * Generate value q_2st*r/M
    gen fdr_temp2 = `qval_2st'*rank/`totalpvals'
    * Generate binary variable checking condition p(r) <= q_2st*r/M
    gen reject_temp2 = (fdr_temp2>=pval) if pval~=.
    * Generate variable containing p-value ranks for all p-values that meet above
      *condition
    gen reject_rank2 = reject_temp2*rank
    * Record the rank of the largest p-value that meets above condition
    egen total_rejected2 = max(reject_rank2)

    * A p-value has been rejected at level q if its rank is less than or equal to
        *the rank of the max p-value that meets the above condition
    replace bky06_qval = `qval' if rank <= total_rejected2 & rank~=.
    * Reduce q by 0.001 and repeat loop
    drop fdr_temp* reject_temp* reject_rank* total_rejected*
    local qval = `qval' - .001
}


quietly sort original_sorting_order
pause off
set more on

display "Code has completed."
display "Benjamini Krieger Yekutieli (2006) sharpened q-vals are in
                                                    variable 'bky06_qval'"
display "Sorting order is the same as the original vector of p-values"

keep outcome pval bky06_qval
save "$output/sharpenedqvals.dta", replace

export delimited using "$output/sharpenedqvals.csv", replace
// guardamos como csv los coef ajustados
```