# Econ 675 Assignment 3

Nathan Mather*

November 25, 2018

## Contents



# 1 Question 1: Many Instruments Asymptotics

## 1.1 Q1 Part 1

The first two results follow immediately while the third follows from $\mathrm{E}[x'u/n] = \mathrm{E}[v'u/n] = \sigma_{uv}$ since Z is non-random. for the ourth result, note that $\mathrm{E}[x'Pu] = \mathrm{E}[v'Pu] = k\sigma_{uv}$ because $\mathrm{E}[v_i u_j] = 0$ for all $i \neq j$ and $\sum_{i=1}^{n} P_{ij} = K$. Finally the last result follows analogously.

## 1.2 q1 Part 2

$$x'x/n = v'v/n + 2v'Z\pi/n + \pi'Z'Z\pi/n \to_p \mu + \sigma_v^2$$

using LLN and because $\mathrm{E}[v'Z\pi/n] = 0$ and $\mathrm{V}[v'Z\pi/n] = O(n^{-1})$. which gives the first result.
For the second result, note that

$$x'Px/n = v'pv/n + 2v'Z\pi/n + \pi'Z'Z\pi/n \to_p \mu + \rho\sigma_v^2$$

*Shouts out to Ani for the help with question 1 which follows from $E[v'Pv/n] = \sum_{i=1}^{n} p_{ii}\sigma_v^2/n = K\sigma_v^2/n \to \rho\sigma_v^2$ , because $\mathrm{E}[v_i v_j] = 0$ for all $i \neq j$, and because $\mathrm{V}[v'pv/n] \to 0$ after some calculations and using basic projection matrices. Specifically,

$$\mathrm{V}[v'Pv/n] = \frac{1}{n^2}\mathrm{V}\left[\sum_{i=1}^{n} p_{ii}v_i^2 + 2\sum_{i<j} p_{ij}v_iv_j\right] = \frac{1}{n^2}\left(\mathrm{V}[v_i^4]\sum_{i=1}^{n} p_{ii}^2 + 4\sigma_v^2\sum_{i<j} p_{ij}^2\right) \leq \frac{C}{n^2} trace(P'P) \leq \frac{CK}{n^2} \to 0$$

Where C is some universal constant greater than zero.

For the third result,

$$x'Pu/n = \pi'Z'u/n + v'Pu/n \to_p 0 + \rho\sigma_{uv}$$

which follows from a similar argument to the second result and part 1.

## 1.3 Q1 part 3

This follows directly by previous results because

$$\hat{\beta}_{2sls} = \beta + (x'Px/n)^{-1}x'Pu/n$$

and the result follows by CMT.

## 1.4 Q1 Part 4

$$\tilde{\beta_{2sls-BC}} = \beta + (x'Px)^{-1}(x'Pu - v'Pu) = \beta + (x'Px)^{-1}\pi'Z'u$$

By the argument to part 2, the second term is $o_p(1)$

## 1.5 Q1 part 5

(a)

$$\begin{aligned}
\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{u} &= (\boldsymbol{\pi}'\boldsymbol{Z}' + \boldsymbol{v}')(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} \\
&= \boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} + \boldsymbol{v}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} \\
&= \boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} + \left(\check{\boldsymbol{v}}' + \frac{\sigma_{uv}^2}{\sigma_u^2}\boldsymbol{u}'\right)(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} \\
&= \boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} + \check{\boldsymbol{v}}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} + \frac{\sigma_{uv}^2}{\sigma_u^2}\boldsymbol{u}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u},
\end{aligned}$$

as required.
(b) Next, note that

$$\mathrm{E}[\boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u}] = \boldsymbol{\pi}'\boldsymbol{Z}'\mathrm{E}[\boldsymbol{u}] - \frac{K}{n}\boldsymbol{\pi}'\boldsymbol{Z}'\mathrm{E}[\boldsymbol{u}] = 0,$$

since $\boldsymbol{Z}$ is nonrandom. Accordingly, the CLT implies that

$$\frac{1}{\sqrt{n}}\boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u} \to_d \mathcal{N}(0, V_1(\rho)),$$

where

$$\begin{aligned}
V_1(\rho) &= \lim_{n\to\infty} \mathrm{V}[1/\sqrt{n}\boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u}] \\
&= \lim_{n\to\infty} \frac{1}{n}\mathrm{E}[\boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{u}\boldsymbol{u}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{Z}\boldsymbol{\pi}] \\
&= \lim_{n\to\infty} \frac{1}{n}\sigma_u^2 \left[\boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)(\boldsymbol{P} - \frac{K}{n}\boldsymbol{I}_n)\boldsymbol{Z}\boldsymbol{\pi}\right] \\
&= \lim_{n\to\infty} \frac{1}{n}\sigma_u^2 \left[\boldsymbol{\pi}'\boldsymbol{Z}'\boldsymbol{Z}\boldsymbol{\pi} - 2\frac{K}{n}\boldsymbol{\pi}'\boldsymbol{Z}'\boldsymbol{Z}\boldsymbol{\pi} + \frac{K^2}{n^2}\boldsymbol{Z}\boldsymbol{\pi}'\boldsymbol{Z}'\boldsymbol{Z}\boldsymbol{\pi}\right] \\
&= \sigma_u^2(1-\rho^2).
\end{aligned}$$

(c) Now,

$$\begin{aligned}
\mathrm{E}[\check{\boldsymbol{v}}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}] &= \mathrm{E}\left[\left(\boldsymbol{v}' - \frac{\sigma_{uv}^2}{\sigma_u^2}\boldsymbol{u}'\right)\boldsymbol{P}\boldsymbol{u} - \frac{K}{n}\left(\boldsymbol{v}' - \frac{\sigma_{uv}^2}{\sigma_u^2}\boldsymbol{u}'\right)\boldsymbol{u}\right] \\
&= \mathrm{E}[\boldsymbol{v}'\boldsymbol{P}\boldsymbol{u}] - \frac{\sigma_{uv}^2}{\sigma_u^2}\mathrm{E}[\boldsymbol{u}'\boldsymbol{P}\boldsymbol{u}] - \frac{K}{n}\mathrm{E}[\boldsymbol{v}'\boldsymbol{u}] + \frac{K}{n}\frac{\sigma_{uv}^2}{\sigma_u^2}\mathrm{E}[\boldsymbol{u}'\boldsymbol{u}]
\end{aligned}$$

Then, plugging in the results from part 1 gives

$$\mathrm{E}[\check{\boldsymbol{v}}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}] = K\sigma_{uv}^2 - \frac{\sigma_{uv}^2}{\sigma_u^2}K\sigma_u^2 - \frac{K}{n}\cdot n\sigma_{uv}^2 + \frac{K}{n}\frac{\sigma_{uv}^2}{\sigma_u^2}\cdot n\sigma_u^2 = 0,$$

as required.

To get the convergence result we would do the following. Compute $\mathrm{V}[\check{\boldsymbol{v}}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}]$. Using the assumption $\mathrm{V}[\boldsymbol{u}|\check{\boldsymbol{v}}] = \sigma_u^2\boldsymbol{I}_n$, it can be shown that

$$\lim_{n\to\infty} \mathrm{V}[\check{\boldsymbol{v}}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}] = O(K).$$

Then, we can somehow use the Markov inequality to get the desired convergence result.
(d) Analogous derivations to the above question give the desired results.
(e) Now,

$$\begin{aligned}
\mathrm{E}[\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{u}] &= \mathrm{E}[(\boldsymbol{\pi}'\boldsymbol{Z}' + \boldsymbol{v}')(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}] \\
&= \mathrm{E}[\boldsymbol{\pi}'\boldsymbol{Z}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}] + \mathrm{E}[\boldsymbol{v}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}] \\
&= 0 + \mathrm{E}[\boldsymbol{v}'\boldsymbol{P}\boldsymbol{u}] - K/n\mathrm{E}[\boldsymbol{v}'\boldsymbol{u}] \\
&= K\sigma_{uv}^2 - K/n \cdot n\sigma_{uv}^2 \\
&= 0.
\end{aligned}$$

And

$$\begin{aligned}
\vartheta^2 = \mathrm{V}[\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{u}/\sqrt{n}] &= \frac{1}{n}\mathrm{E}[\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{u}\boldsymbol{u}'\check{\boldsymbol{P}}\boldsymbol{x}] \\
&= \frac{1}{n}\mathrm{E}[\boldsymbol{x}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{u}\boldsymbol{u}'(\boldsymbol{P} - K/n\boldsymbol{I}_n)\boldsymbol{x}] \\
&= \frac{1}{n}\mathrm{E}[(\boldsymbol{x}'\boldsymbol{P}\boldsymbol{u} - K/n\boldsymbol{x}'\boldsymbol{u})(\boldsymbol{u}'\boldsymbol{P}\boldsymbol{x} - K/n\boldsymbol{u}'\boldsymbol{x})]
\end{aligned}$$

(f) Note that

$$\sqrt{n}(\hat{\beta}_{\mathtt{2SLS}} - \beta) = (\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{x}/n)^{-1}(\frac{1}{\sqrt{n}}\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{u})$$

And we assume that

$$\frac{1}{\sqrt{n}}\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{u} \to_d \mathcal{N}(0, \vartheta^2)$$

Thus,

$$\sqrt{n}(\hat{\beta}_{\mathtt{2SLS}} - \beta) \to_d \mathcal{N}(0, \mathrm{E}[\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{x}]^{-1}\vartheta^2\mathrm{E}[\boldsymbol{x}'\check{\boldsymbol{P}}\boldsymbol{x}]^{-1})$$

## 2 Question 2: Weak Instruments Simulations

**Results for $n\gamma^2 = 0$**

| reg_type | variable | mean | st.dev | quant .1 | quant .5 | quant .9 |
|---|---|---|---|---|---|---|
| ols | estimate | 1.00 | 0.01 | 0.99 | 1.00 | 1.01 |
| ols | std.error | 0.01 | 0.00 | 0.01 | 0.01 | 0.01 |
| ols | rej | 1.00 | 0.00 | 1.00 | 1.00 | 1.00 |
| 2sls | estimate | 0.66 | 20.76 | 0.68 | 1.00 | 1.32 |
| 2sls | std.error | 3248.34 | 182231.00 | 0.07 | 0.22 | 4.95 |
| 2sls | rej | 0.69 | 0.46 | 0.00 | 1.00 | 1.00 |
| 2sls | f_stat | 1.00 | 1.39 | 0.01 | 0.44 | 2.65 |

**Results for $n\gamma^2 = 0.25$**

| reg_type | variable | mean | st.dev | quant .1 | quant .5 | quant .9 |
|---|---|---|---|---|---|---|
| ols | estimate | 1.00 | 0.01 | 0.99 | 1.00 | 1.01 |
| ols | std.error | 0.01 | 0.00 | 0.01 | 0.01 | 0.01 |
| ols | rej | 1.00 | 0.00 | 1.00 | 1.00 | 1.00 |
| 2sls | estimate | 0.28 | 31.08 | -0.97 | 0.65 | 2.64 |
| 2sls | std.error | 1630.89 | 91246.48 | 0.15 | 0.89 | 23.65 |
| 2sls | rej | 0.32 | 0.47 | 0.00 | 0.00 | 1.00 |
| 2sls | f_stat | 1.26 | 1.81 | 0.02 | 0.57 | 3.44 |

**Results for $n\gamma^2 = 9$**

| reg_type | variable | mean | st.dev | quant .1 | quant .5 | quant .9 |
|---|---|---|---|---|---|---|
| ols | estimate | 0.96 | 0.02 | 0.94 | 0.96 | 0.98 |
| ols | std.error | 0.02 | 0.00 | 0.01 | 0.02 | 0.02 |
| ols | rej | 1.00 | 0.00 | 1.00 | 1.00 | 1.00 |
| 2sls | estimate | -0.31 | 6.73 | -0.77 | -0.01 | 0.29 |
| 2sls | std.error | 15.57 | 713.82 | 0.17 | 0.34 | 1.06 |
| 2sls | rej | 0.08 | 0.27 | 0.00 | 0.00 | 0.00 |
| 2sls | f_stat | 9.99 | 6.34 | 2.83 | 8.88 | 18.34 |

**Results for $n\gamma^2 = 99$**

| reg_type | variable | mean | st.dev | quant .1 | quant .5 | quant .9 |
|---|---|---|---|---|---|---|
| ols | estimate | 0.67 | 0.03 | 0.62 | 0.67 | 0.71 |
| ols | std.error | 0.03 | 0.00 | 0.03 | 0.03 | 0.04 |
| ols | rej | 1.00 | 0.00 | 1.00 | 1.00 | 1.00 |
| 2sls | estimate | -0.01 | 0.11 | -0.15 | -0.00 | 0.11 |
| 2sls | std.error | 0.10 | 0.02 | 0.08 | 0.10 | 0.14 |
| 2sls | rej | 0.05 | 0.21 | 0.00 | 0.00 | 0.00 |
| 2sls | f_stat | 100.93 | 24.69 | 71.05 | 99.09 | 133.35 |

Stata is terrible at putting things into tex and it's extremely tedious to do it by hand, but the results are comparable and the code is in the appendix.

Weak instruments make it difficult if not impossible to infer anything from our estimates. The standard deviation of the estimate and corresponding standard errors are huge. I beleive we showed in 672 that as an instroment becomes weaker, the finite sample distribution approaches cauchy. Even in the case of a very weak instrument, as a sample size goes to infinity it the estimate will become unbiased. However, this may be so slow as to be unreasonable with any realistic sample size.

# 3 Question 3: Weak Instrument - Empirical Study

## 3.1 Question 3.1

**Results from R**

| model | term | estimate | std.error |
|---|---|---|---|
| OLS 1 | educ | 0.06 | 0.00 |
| OLS 2 | educ | 0.06 | 0.00 |
| 2sls 1 | educ | 0.09 | 0.02 |
| 2sls 2 | educ | 0.06 | 0.03 |

In the absence of the weak instrument issues this could be interpreted as a causal relationship where education causes higher earnings. However, as we show below, the instrument is not very good and so these results are essentially meaningless.

## 3.2 Question 3.2

**Results from R**

| model | mean | std.dev |
|---|---|---|
| 2sls 1 | 0.06 | 0.04 |
| 2sls 2 | 0.06 | 0.04 |

These results show us that weak instruments can be a serious problem and lead to results that are completely incorrect. moreover, as we see from the standard errors and standard deviations, std.errors will not appropriately capture the level of uncertainty that arises from a weak instrument.

# 4 Appendix

## 4.1 R Code

# pset 5 675

```r
#====================#
# ==== ps_5_675R ====
#====================#


#==========================================#
# ==== Load packages, clear workspace ====
#==========================================#


library(MASS)
library(data.table)
library(broom)
library(AER)
library(xtable)
library(Matrix)
library(doParallel)
library(foreach)

rm(list = ls(pos = ".GlobalEnv"), pos = ".GlobalEnv")
options(scipen = 999)
cat("\f")

#========================#
# ==== Q2 simulation ====
#========================#

# set final run parm for n simulations and if it should save
final_run <- TRUE


  #=============================#
  # ==== Write sim function ====
  #=============================#

    # parms for funciton
    f_stat = 0
    n = 200
    sim = 1 # a tag for the simulation number

    # sim function
    sim_fun2 <- function(sim, f_stat, n = 200){

      # make gamma
      gamma <- sqrt(f_stat/n)

     # make mu vector
      mu = c(0,0,0)

      # make sigma matrix
```

```r
sigma <- matrix(c(1,0,0,0,1,.99,0,1,.99), 3,3)

# make data
rdt <- mvrnorm(n, mu, sigma)

# make it a data.table
rdt <- data.table(rdt)
setnames(rdt, colnames(rdt), c("z","u","v"))

# back out x
rdt[, x := gamma*z + v]

# back out y given b=0
rdt[, y := u]

# run ols
ols_res <- data.table(tidy(lm(y~x, data = rdt)))

# make column of rejecting the null
ols_res[, rej := as.numeric(abs(statistic) > 1.96)]

# take what we need
ols_res <- ols_res[term == "x", c("estimate", "std.error", "rej")]

# add on ols suffix
ols_res[, reg_type := "ols"]

# melt data for matias table
ols_res <- melt.data.table(id.vars = "reg_type", data = ols_res)

# run first stage of 2sls to get f test
fst_stg <- lm(x~z, data = rdt)
f_stat <- summary(fst_stg)$fstatistic[1]

# now run 2sls
iv_reg <- ivreg(y ~ x | z , data = rdt)
summary(iv_reg)
iv_reg <- data.table(tidy(ivreg(y ~ x | z , data = rdt)))

# compute rej
iv_reg[, rej := as.numeric(abs(statistic) > 1.96)]

# take what we need
iv_reg <- iv_reg[term == "x", c("estimate", "std.error", "rej")]

# throw in the f stat
iv_reg[, f_stat := f_stat]

# add 2sls indicator
iv_reg[, reg_type := "2sls"]

# melt data for matias table
iv_reg <- melt.data.table(id.vars = "reg_type", data = iv_reg)
```

```r
    # stack  these tables
    out_dt <- rbind(ols_res, iv_reg)

    # add sim number
    out_dt[, sim := sim]

    # ruturn that shiz
    return(out_dt[])

    }# end sim funciton




#===========================#
# ==== run sim funciton ====
#===========================#

  # time this sucker
  start_time <- Sys.time()

  # initialize list to store output
  sim_list <- list()
  for(f_stat_i in c(0,.25,9,99)){

    # get number of sims
    n_sims <- ifelse(final_run, 5000, 50)

    # apply the function 5000 times
    sim_out <- lapply(c(1:n_sims), sim_fun2, f_stat = f_stat_i, n = 200)

    # bind the results
    sim_out <- rbindlist(sim_out)

    # take mean, std, quantiles by group
    results <- sim_out[, list("mean" = mean(value),
                              "st.dev" = sd(value),
                              "quant .1" = quantile(value, .1),
                              "quant .5" = quantile(value, .5),
                              "quant .9" = quantile(value, .9)), by = c("reg_type", "variable")]

    # store results in a list
    sim_list[[paste0(f_stat_i)]] <- results

  }# end loop over gamams


  # check time
  end_time <- Sys.time()

  # print time
  print(paste0(round(as.numeric(end_time - start_time, units = "mins"), 3), " minutes to run"))
```

```
#==================================================#
# ==== save out these tables into a tex file ====
#==================================================#

  # check if this is a final run
  if(final_run){

    # for each item in the list
    for(tab_i in ls(sim_list)){

      print(xtable(sim_list[[tab_i]], type = "latex"),
            file = paste0("C://Users/Nmath_000/Documents/Code/courses/econ 675/PS_5_tex/q2tab_fstat_"
            include.rownames = FALSE,
            floating = FALSE)
      }#end loop

  } # end if statement



#=====================#
# ==== Question 3 ====
#=====================#

  # clear enviroment
  rm(list = ls(pos = ".GlobalEnv"), pos = ".GlobalEnv")

  # load in data
  ak <- fread("C:/Users/Nmath_000/Documents/MI_school/Second Year/675 Applied Econometrics/hw/hw5/Angri


  #========================#
  # ==== 3.1 AK models ====
  #========================#

    #============================#
    # ==== regression set up ====
    #============================#


    # make YOB dummies
    ak[, .N, "YoB_ld"]
    for(year_i in unique(ak$YoB_ld)){

      ak[ ,temp := 0]
      ak[YoB_ld == year_i ,temp := 1]
      setnames(ak, "temp", paste0("d_YOB_ld_", year_i))

    }
    # get a list of all year dummies but one. Exclude the proper one to match coeffs
    year_dummies <- setdiff(grep("d_YOB", colnames(ak), value = TRUE), "d_YOB_ld_0")

    # make QoB dummies
```

```r
for(qob_i in unique(ak$QoB)){

  ak[ ,temp := 0]
  ak[QoB == qob_i ,temp := 1]
  setnames(ak, "temp", paste0("d_QoB_", qob_i))

}

# get qob dummy list.  Exclude the proper one to match coeffs
qob_dummies <- setdiff(grep("d_QoB", colnames(ak), value = TRUE), "d_QoB_1")

# make cross variables of year dummies and qob
#note there is almost certainly a better way to do this but here we are
inter_list <- NULL
for(d_year in year_dummies){

  for(d_qob in qob_dummies){

    ak[, temp := get(d_qob)*get(d_year)]
    setnames(ak, "temp", paste0(d_year, "X", d_qob))
    inter_list<- c(inter_list, paste0(d_year, "X", d_qob))
  }
}


# standard controls
# (i) race, (ii) marrital status, (iii) SMSA, (iv) dummies for
# region, and (iv) dummies for YoB ld.
std_cont <- c("non_white","married", "SMSA",
              "ENOCENT","ESOCENT", "MIDATL",
              "MT", "NEWENG", "SOATL", "WNOCENT",
              "WSOCENT",  year_dummies) # get year dummies but leave one out

# save extra controls
extra_cont <- c("age_q", "age_sq")

#=================#
# ==== ols 1 ====
#=================#

  # make the formula
  ols1_form <- as.formula(paste0("l_w_wage~educ +", paste(std_cont, collapse = " + ")))

  # run ols
  out_ols1 <- data.table(tidy(lm(ols1_form, data = ak)))

  # keep what I need
  out_ols1 <- out_ols1[term %chin% c("educ"), c("term", "estimate", "std.error")]
  out_ols1[, model := "OLS 1"]

#=================#
# ==== OlS 2 ====
#=================#
```

```
  # make the formula
  ols2_form <- as.formula(paste0("l_w_wage~educ +", paste(std_cont, collapse = " + "), " + ", paste0

  #run ols
  out_ols2 <- data.table(tidy(lm(ols2_form, data = ak)))

  # keep what I need
  out_ols2 <- out_ols2[term %chin% c("educ"), c("term", "estimate", "std.error")]
  out_ols2[, model := "OLS 2"]

#================#
# ==== 2sls ====
#================#

  # write this part as a function so I can use it in 3.2
  # ACTUALLY, im gonna use different faster function but this is fine as a function too
  wrap_2sls <- function(in_data){

  #==================#
  # ==== 2sls 1 ====
  #==================#

    iv_form <- as.formula(paste0("l_w_wage~educ +", paste(std_cont, collapse = " + "),
                                 "| ",
                                 paste(std_cont, collapse = " + "), " + ", paste0(inter_list, colla
    iv_reg1 <- data.table(tidy(ivreg(iv_form , data = in_data)))

    # keep what I need
    iv_reg1 <- iv_reg1[term %chin% c("educ"), c("term", "estimate", "std.error")]
    iv_reg1[, model := "2sls 1"]

  #==================#
  # ==== 2sls 2 ====
  #==================#


    iv_form2 <- as.formula(paste0("l_w_wage~educ +", paste(std_cont, collapse = " + "), "+", paste0
                                 "| ",
                                 paste(std_cont, collapse = " + "),
                                 " + ", paste0(inter_list, collapse = " + "),
                                 "+", paste0(extra_cont, collapse = " + ")))
    iv_reg2 <- data.table(tidy(ivreg(iv_form2 , data = in_data)))

    # keep what I need
    iv_reg2 <- iv_reg2[term %chin% c("educ"), c("term", "estimate", "std.error")]
    iv_reg2[, model := "2sls 2"]

    # stack 2sls
    out_2sls <- rbind(iv_reg1, iv_reg2)

    return(out_2sls)

  }#end 2sls function
```

```
    # run function
    ak_2sls <- wrap_2sls(ak)

  #========================#
  # ==== output tables ====
  #========================#

    output_3.1 <- rbind(out_ols1, out_ols2, ak_2sls)
    setcolorder(output_3.1, c("model", "term", "estimate", "std.error"))

    # out put it

    print(xtable(output_3.1, type = "latex"),
          file = paste0("C://Users/Nmath_000/Documents/Code/courses/econ 675/PS_5_tex/q3.1_table.tex")
          include.rownames = FALSE,
          floating = FALSE)



#================#
# ==== Q 3.2 ====
#================#

  #=========================================#
  # ==== whats the fastest 2sls method? ====
  #=========================================#

  # #================#
  # # ==== ivreg ====
  # #================#
  #
  #
  #
  #   # using ivreg
  # start1 <- Sys.time()
  #
  #   iv_form <- as.formula(paste0("l_w_wage~educ +", paste(std_cont, collapse = " + "),
  #                                "| ",
  #                                paste(std_cont, collapse = " + "), " + ", paste0(inter_list, colla
  #   iv_reg1 <- data.table(tidy(ivreg(iv_form , data = ak_perm)))
  #
  # end1 <- Sys.time()
  # print(paste0(round(as.numeric(end1 - start1, units = "secs"), 3), " seconds to run"))
  #
  # #=========================#
  # # ==== using matrix ====
  # #=========================#
  #
  #   ak_perm[, const := 1]
  #   start1 <- Sys.time()
  # # make x z and y matrices
  # y <- as.matrix(ak_perm[, l_w_wage])
  # x <- as.matrix(ak_perm[, c("educ", std_cont, 'const'), with = FALSE])
```

```
# z <- as.matrix(ak_perm[, c(inter_list, std_cont, "const"), with = FALSE])
#
# # get 2sls
# out_2sls1 <- solve(crossprod(x,z)%*%solve(crossprod(z))%*%crossprod(z,x))%*%crossprod(x,z)%*%solv
# end1 <- Sys.time()
# print(paste0(round(as.numeric(end1 - start1, units = "secs"), 3), " seconds to run"))
#
# #===================#
# # ==== using lm ====
# #===================#
#
# start1 <- Sys.time()
#
# form_1st <- as.formula(paste0("educ~", paste(std_cont, collapse = " + "), " + ", paste0(inter_lis
# first_stage <- lm(form_1st, data = ak_perm)
#
#
# X_hat <- fitted(first_stage)
# form_2nd <- as.formula(paste0("l_w_wage~"," X_hat +", paste(std_cont, collapse = " + ")))
#
# ols_second <- lm(form_2nd, data = ak_perm)
# coef(ols_second)
# end1 <- Sys.time()
# print(paste0(round(as.numeric(end1 - start1, units = "secs"), 3), " seconds to run"))
#
#
#=======================#
# ==== matrix try 2 ====
#=======================#

# # LOOKS LIKE THIS IS THE WAY TO GO
# start1 <- Sys.time()
#
# # make x z and y matrices
# y <- as.matrix(ak_perm[, l_w_wage])
# x <- as.matrix(ak_perm[, educ])
# cont <- as.matrix(ak_perm[, c( std_cont, 'const'), with = FALSE])
# z <- as.matrix(ak_perm[, c(inter_list, std_cont, "const"), with = FALSE])
#
# first_stage_fit <-  z%*%Matrix::solve(Matrix::crossprod(z))%*%(Matrix::crossprod(z, x))
#
# # make x' matrix
# x_prime <- cbind(first_stage_fit, cont)
#
#
# form_2nd <-  Matrix::solve(Matrix::crossprod(x_prime))%*%(Matrix::crossprod(x_prime, y))
#
# form_2nd[1,1]
# end1 <- Sys.time()
# print(paste0(round(as.numeric(end1 - start1, units = "secs"), 3), " seconds to run"))
#
#===================================#
# ==== write fast 2sls function ====
```

```r
  #====================================#

  fast_2sls <- function(in_data){

    #================#
    # ==== reg1 ====
    #================#

    # make x z and y matrices
    y <- as.matrix(in_data[, l_w_wage])
    x <- as.matrix(in_data[, educ])
    cont <- as.matrix(in_data[, c( std_cont, 'const'), with = FALSE])
    z <- as.matrix(in_data[, c(inter_list, std_cont, "const"), with = FALSE])

    first_stage_fit <-  z%*%Matrix::solve(Matrix::crossprod(z))%*%(Matrix::crossprod(z, x))

    # make x' matrix
    x_prime <- cbind(first_stage_fit, cont)


    form_2nd <-  Matrix::solve(Matrix::crossprod(x_prime))%*%(Matrix::crossprod(x_prime, y))

    reg1 <- data.table( term = "educ", estimate = form_2nd[1,1], model = "2sls 1")

    #================#
    # ==== reg2 ====
    #================#

    cont <- as.matrix(in_data[, c( std_cont, extra_cont, 'const'), with = FALSE])
    z <- as.matrix(in_data[, c(inter_list, std_cont, extra_cont, "const"), with = FALSE])

    first_stage_fit <-  z%*%Matrix::solve(Matrix::crossprod(z))%*%(Matrix::crossprod(z, x))

    # make x' matrix
    x_prime <- cbind(first_stage_fit, cont)


    form_2nd <-  Matrix::solve(Matrix::crossprod(x_prime))%*%(Matrix::crossprod(x_prime, y))

    reg2 <- data.table( term = "educ", estimate = form_2nd[1,1], model = "2sls 2")

    # stack results and retur n
    out_results <- rbind(reg1, reg2)
  }


#==========================#
# ==== run simulation ====
#==========================#

  # copy data for permutation
  ak_perm <- copy(ak)
```

```r
# add constant
ak_perm[, const := 1]

# write a function so I can parallel this shiz
sim_warper <- function(sim_i, in_data = ak_perm ){

  # get random sampel
  perm <- sample(c(1:nrow(in_data)))

  # purmute data
  in_data[, QoB := QoB[perm]]

  # clear out dummy variables
  in_data <- in_data[, -c(grep("d_QoB", colnames(in_data), value = TRUE), inter_list), with = FALSE]

  # redo dummy vars
  for(qob_i in unique(in_data$QoB)){

    in_data[ ,temp := 0]
    in_data[QoB == qob_i ,temp := 1]
    setnames(in_data, "temp", paste0("d_QoB_", qob_i))

  }
  # recalculate interactions
  inter_list <- NULL
  for(d_year in year_dummies){

    for(d_qob in qob_dummies){

      in_data[, temp := get(d_qob)*get(d_year)]
      setnames(in_data, "temp", paste0(d_year, "X", d_qob))
      inter_list<- c(inter_list, paste0(d_year, "X", d_qob))
    }
  }

  # run 2sls funciton on new data
  ak_2sls_i <- fast_2sls(in_data)

  # add simulation
  ak_2sls_i[, sim := sim_i]

  # retunrn it
  return(ak_2sls_i)

} # end funciton

# time this sucker
start_time <- Sys.time()

# parallel setup
cl <- makeCluster(4, type = "PSOCK")
registerDoParallel(cl)
```

```r
# run simulations in parallel
output_list <- foreach(sim = 1 : 5000,
                       .inorder = FALSE,
                       .packages = "data.table",
                       .options.multicore = list(preschedule = FALSE, cleanup = 9)) %dopar% sim_war

# stop clusters
stopCluster(cl)

# check time
end_time <- Sys.time()

# print time
print(paste0(round(as.numeric(end_time - start_time, units = "mins"), 3), " minutes to run"))


#==========================#
# ==== organize output ====
#==========================#

  # stack data
  sim_res3.2 <- rbindlist(output_list)

  # make table
  output3.2 <- sim_res3.2[, list(mean = mean(estimate), std.dev = sd(estimate)), "model"]

  # save it

  print(xtable(output3.2, type = "latex"),
        file = paste0("C://Users/Nmath_000/Documents/Code/courses/econ 675/PS_5_tex/q3.2_table.tex"
        include.rownames = FALSE,
        floating = FALSE)
```

### 4.2 STATA Code

```

*********************************************************************
* Question 2
*********************************************************************
clear all
set more off
cap log close

program define weak_IV, rclass
    syntax [, obs(integer 200) f_stat(real 10) ]
    drop _all

    set obs `obs'

    * DGP
    gen u = rnormal()
    gen v = 0.99 * u + sqrt(1-0.99^2) * rnormal()
    gen z = rnormal()

    local gamma_0 = sqrt((`f_stat' - 1) / `obs')
    gen x = `gamma_0' * z + v
    gen y = u

    * OLS
    qui reg y x, robust
    return scalar OLS_b   = _b[x]
    return scalar OLS_se  = _se[x]
    return scalar OLS_rej = abs(_b[x]/_se[x]) > 1.96

    * 2SLS
    qui ivregress 2sls y (x = z)
    return scalar TSLS_b   = _b[x]
    return scalar TSLS_se  = _se[x]
    return scalar TSLS_rej = abs(_b[x]/_se[x]) > 1.96
    qui reg x z
    return scalar TSLS_F   = e(F)
end

* simulation 1: F = 1
simulate OLS_b=r(OLS_b) OLS_se=r(OLS_se) OLS_rej=r(OLS_rej) ///
    TSLS_b=r(TSLS_b) TSLS_se=r(TSLS_se) TSLS_rej=r(TSLS_rej) TSLS_F=r(TSLS_F), ///
    reps(5000) seed(123) nodots: ///
    weak_IV, f_stat(1)

local k = 1
matrix Results = J(7, 5, .)

qui sum OLS_b, detail
matrix Results[`k',1] = r(mean)
matrix Results[`k',2] = r(sd)
matrix Results[`k',3] = r(p10)
matrix Results[`k',4] = r(p50)
matrix Results[`k',5] = r(p90)
local k = `k' + 1

qui sum OLS_se, detail
matrix Results[`k',1] = r(mean)
matrix Results[`k',2] = r(sd)
matrix Results[`k',3] = r(p10)
matrix Results[`k',4] = r(p50)
matrix Results[`k',5] = r(p90)
local k = `k' + 1

qui sum OLS_rej, detail
matrix Results[`k',1] = r(mean)
matrix Results[`k',2] = r(sd)
matrix Results[`k',3] = r(p10)
matrix Results[`k',4] = r(p50)
matrix Results[`k',5] = r(p90)
local k = `k' + 1
```

```

qui sum TSLS_b, detail
matrix Results[`k',1] = r(mean)
matrix Results[`k',2] = r(sd)
matrix Results[`k',3] = r(p10)
matrix Results[`k',4] = r(p50)
matrix Results[`k',5] = r(p90)
local k = `k' + 1

qui sum TSLS_se, detail
matrix Results[`k',1] = r(mean)
matrix Results[`k',2] = r(sd)
matrix Results[`k',3] = r(p10)
matrix Results[`k',4] = r(p50)
matrix Results[`k',5] = r(p90)
local k = `k' + 1

qui sum TSLS_rej, detail
matrix Results[`k',1] = r(mean)
matrix Results[`k',2] = r(sd)
matrix Results[`k',3] = r(p10)
matrix Results[`k',4] = r(p50)
matrix Results[`k',5] = r(p90)
local k = `k' + 1

qui sum TSLS_F, detail
matrix Results[`k',1] = r(mean)
matrix Results[`k',2] = r(sd)
matrix Results[`k',3] = r(p10)
matrix Results[`k',4] = r(p50)
matrix Results[`k',5] = r(p90)
local k = `k' + 1

mat2txt, matrix(Results) saving(result1.txt) format(%9.4f) replace



*******************************************************************
* Question 3
*******************************************************************
clear all
set more off
cap log close
use "Angrist_Krueger.dta"

*************************************************************************
* The following replicates Columns (5)-(8), Table V
* in Angrist and Krueger (1991 QJE)
*************************************************************************

*** Column 5, Table V, Angrist and Krueger (1991 QJE)
reg l_w_wage educ non_white married SMSA i.region i.YoB_ld

*** Column 6, Table V, Angrist and Krueger (1991 QJE)
ivregress 2sls l_w_wage non_white married SMSA i.region i.YoB_ld ///
    (educ = i.YoB_ld##i.QoB)
estat firststage

*** Column 7, Table V, Angrist and Krueger (1991 QJE)
reg  l_w_wage educ non_white married SMSA age_q age_sq i.region i.YoB_ld

*** Column 8, Table V, Angrist and Krueger (1991 QJE)
ivregress 2sls l_w_wage non_white married SMSA age_q age_sq i.region i.YoB_ld ///
    (educ = i.YoB_ld##i.QoB)
estat firststage

*************************************************************************
* The following replicates Columns (1) and (2), Table 3
* in Bound et al. (1995)
*************************************************************************
```

```
capture program drop IV_quick
program define IV_quick, rclass
    syntax varlist(max=1) [, model(integer 1) ]
    local x "`varlist'"

    if (`model' == 1) {
        capture drop educ_hat
        qui reg educ non_white married SMSA i.region i.YoB_ld i.YoB_ld##i.`x'
        predict educ_hat
        qui reg l_w_wage educ_hat non_white married SMSA i.region i.YoB_ld
        return scalar beta = _b[educ_hat]
    }
    if (`model' == 2) {
        capture drop educ_hat
        qui reg educ non_white married SMSA age_q age_sq i.region i.YoB_ld i.YoB_ld##i.`x'
        predict educ_hat
        qui reg l_w_wage educ_hat non_white married SMSA age_q age_sq i.region i.YoB_ld
        return scalar beta = _b[educ_hat]
    }
end


permute QoB TSLS_1_b = r(beta), reps(500) seed(123) saving(premute1, replace): ///
    IV_quick QoB, model(1)

permute QoB TSLS_2_b = _b[educ], reps(500) seed(123) saving(premute2, replace): ///
    ivregress 2sls l_w_wage non_white married SMSA age_q age_sq i.region i.YoB_ld ///
    (educ = i.YoB_ld##i.QoB)

clear all
use "premute1.dta"
sum TSLS_1_b

clear all
use "premute2.dta"
sum TSLS_2_b
```